*NỀN TẢNG QUẢN TRỊ DOANH NGHIỆP TỔNG THỂ*

*KIẾN TRÚC PHẦN MỀM*

|  |  |
| --- | --- |
| Ngày: | 24/02/2023 |
| Phiên bản: | 1.0.0 |

**LỊCH SỬ THAY ĐỔI**

|  |  |  |  |
| --- | --- | --- | --- |
| Phiên bản | Người thực hiện | Ngày sửa đổi | Lý do thay đổi |
| 1.0.0 | Phạm Đức Chính | 19/12/2023 | Khởi tạo tài liệu |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**LỊCH SỬ PHÊ DUYỆT**

|  |  |  |  |
| --- | --- | --- | --- |
| Phiên bản | Người phê duyệt | Ngày phê duyệt | Ghi chú |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**MỤC LỤC**

[I. TỔNG QUAN SẢN PHẨM 4](#_Toc167460130)

[1. Giới thiệu sản phẩm 4](#_Toc167460131)

[2. Đối tượng sử dụng sản phẩm 4](#_Toc167460132)

[3. Công nghệ sử dụng trong sản phẩm 4](#_Toc167460133)

[II. KIẾN TRÚC 7](#_Toc167460134)

[1. Ngữ cảnh của sản phẩm 7](#_Toc167460135)

[2. Kiến trúc tổng quan 8](#_Toc167460136)

[2.1. Hệ thống SSO 11](#_Toc167460137)

[2.2. Caching 12](#_Toc167460138)

[2.3 Logging 13](#_Toc167460139)

[2.4 Microservice communication 14](#_Toc167460140)

[2.5 Distributed Transaction 15](#_Toc167460141)

[3. Kiến trúc triển khai 17](#_Toc167460142)

[4. Kiến trúc Multi-tenancy (SaaS) 18](#_Toc167460143)

[III. TIÊU CHUẨN CHẤT LƯỢNG 20](#_Toc167460144)

[1. Hiệu năng 20](#_Toc167460145)

[2. Tính khả dụng 20](#_Toc167460146)

[3. Tính an toàn – bảo mật 21](#_Toc167460147)

[4. Tính sửa chữa và bảo trì 28](#_Toc167460148)

[5. Tính tự kiểm thử 28](#_Toc167460149)

[6. Tính mở rộng 28](#_Toc167460150)

[7. Tính tái sử dụng 28](#_Toc167460151)

[8. Tính hỗ trợ 29](#_Toc167460152)

[9. Tính tích hợp 29](#_Toc167460153)

[10. Tính khôi phục thảm họa 29](#_Toc167460154)

[PHỤ LỤC I: CÁC YẾU TỐ ẢNH HƯỞNG 29](#_Toc167460155)

[1. Yếu tố về thời gian và ngân sách 29](#_Toc167460156)

[2. Yếu tố về công nghệ 29](#_Toc167460157)

[3. Yếu tố về nhân sự 30](#_Toc167460158)

# I. TỔNG QUAN SẢN PHẨM

## Giới thiệu sản phẩm

Nền tảng quản trị doanh nghiệp tổng thể là hệ sinh thái doanh nghiệp số, cung cấp đa dạng Các sản phẩm đáp ứng tính năng cơ bản chưa có nhiều các tính năng chuyên sâu và thông minh. Việc cập nhật tính năng sản phẩm theo yêu cầu của thị trường, theo sự thay đổi chính sách của cơ quan quản lý nhà nước còn chậm. Các sản phẩm tập trung theo 4 mảng chính:

• Các phần mềm cho quản trị doanh nghiệp: Kế toán; Quản lý tài sản; quản lý nhân lực; CRM; Loyalty

• Các phần mềm sản phẩm chuyên sâu: quản lý trung tâm đào tạo; quản lý trung tâm sửa chữa/bảo dưỡng; quản lý nhà hàng; quản lý khách sạn; quản lý kênh phân phối

• Các dịch vụ tiện ích: Hóa đơn; CA; email; hosting; …

• Các dịch vụ hạ tầng - viễn thông: Di động; băng rộng cố định

## Đối tượng sử dụng sản phẩm

|  |  |
| --- | --- |
| Đối tượng sử dụng | Tương tác |
| Người dùng | Thao tác qua giao diện web, mobile app (android, iOS) |

## Công nghệ sử dụng trong sản phẩm

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Thành phần | Công nghệ | Phiên bản |
| Hệ điều hành | Linux | Ubuntu |
| Nền tảng lập trình Back-end | .Net | .Net 8 |
| Nền tảng lập trình Di động | Flutter | 3.7.6 |
| Nền tảng lập trình Front-end | Razor | .Net 8 |
| Hệ quản trị CSDL | MySQL, MongoDB | MySQL: 8.0  MongoDB: 6.0 |
| API Gateway | Ocelot | 22.0.1 |
| Messaging và Streaming | Kafka (IDG) | 2.7.0 |
| Caching | Redis (IDG) | 6.2.5 |
| Lưu trữ | MinIO (IDG) | 2.2.0 |
| Proxy và Cân bằng tải | Nginx | 1.23.3 |
| Giám sát và Cảnh báo | Prometheus, Grafana | Prometheus: v2.22.1  Grafana: 8.3.3 |
| Logging | Fluentd | 1.15.2 |
| Đóng gói | Docker | 19.03.13 |
| Triển khai | K8s | 1.19.3 |

# II. KIẾN TRÚC

## 1. Ngữ cảnh của sản phẩm

A diagram of a flowchart

Description automatically generated

Hệ thống HST SME HDDT có tương tác với các hệ thống bên ngoài cụ thể như sau:

IDG: Sử dụng dịch vụ nền tảng của IDG (Redis, Kafka, Logging và MinIO), qua UI portal của IDG và SDK được cung cấp.

Hệ thống ngoài: Các hệ thống tích hợp với HST SME HDDT theo giao tiếp Resful api, Kafka để trao đổi dữ liệu.

## 2. Kiến trúc tổng quan

A diagram of a chat

Description automatically generated

Mô tả các thành phần của hệ thống

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên module** | **Mô tả** |
| 1 | Web UI | Phần web client là các web nơi chứa giao diện cho người sử dụng. Ở đây bao gồm 2 module: Web giao diện cho người dùng admin dùng để quản trị hệ thống và phần web cho khách hàng. Web client sử dụng razor page của nền tảng ASP.NET Core và sử dụng các control của thư viện Devextreme (Devexpress) để xây dựng form và các report. Do hệ thống được xây dựng để tích hợp với các hệ thống có sẵn HKD, ASME, BMIS, PointPlus, … với dao diện đồng nhất nên sử dụng chung công nghệ với các dự án cũ đang dùng. |
| 2 | Mobile app | Ứng dụng trên mobile của hệ thống OnePlace. Được phát triển trên nền tảng Flutter cung cấp ra giao diện hệ thống trên nền tảng Android và IOS. |
| 3 | **API Gateway** | Là một ứng dụng dạng microservice chạy trên nền docker. API Gateway sử dụng trên nền tảng công nghệ Ocelot gateway. Ứng dụng đóng vai trò như một router định tuyến tất cả các request từ phía client đến các service được yêu cầu. Tất cả các kết nối từ bên ngoài hệ thống đều phải đi qua API Gateway mà không thể kết nối trực tiếp vào các service bên trong. API Gateway sẽ cập nhật và public tất cả các API của các service mà nó kết nối ra môi trường bên ngoài. Cụ thể ở trong hệ thống hiện tại API Gateway kết nối với các service backend đằng sau để cung cấp các API theo nghiệp vụ, đóng vai trò trung gian khi các request từ bên ngoài kết nối với 2 service bên trong. Ngoài ra API Gateway còn đóng vai trò cân bằng tải, xác thực request cũng như định tuyến các request vào hệ thống. |
| 4 | **Identity Service** | Là một service cung cấp các API cho việc xác thực người dùng, quản lý người dùng, quản lý đơn vị, quản lý dịch vụ. Hệ thống được xác thực API dựa trên Json Web Token (JWT). Identity service là nơi cung cấp API để đăng nhập vào hệ thống đồng thời tạo ra JWT và trả lại cho phía client. Client mỗi khi request tới hệ thống đều phải kèm theo JWT để xác thực request có hợp lệ hay không. |
| 5 | **Invoice Service** | Là service nơi cung cấp ra các API xử lý các nghiệp vụ liên quan đến hóa đơn điện tử. |
| 6 | **Redis DB** | Là một NonSQL DB, lưu dữ liệu dưới dạng cặp key-value. Redis có tốc độ truy suất cực cao do dữ liệu được ghi trên RAM. Redis được dùng trong hệ thống với mục đích dùng để caching, lưu trữ các thông tin cần truy xuất nhanh. Chia sẽ session giữa các services. |
| 7 | **MySql** | Database được sử dụng trong hệ thống là MySQL. Là nơi chứa tất cả các dữ liệu của hệ thống. |
| 8 | **MinIO** | Là server lưu trữ file dạng Object Storage được cung cấp trên nền tảng hạ tầng IDG. Hệ thống sử dụng để lưu các file dữ liệu upload từ phía người dùng. |
| 9 | **MongoDB** | Là cơ sỡ dữ liệu NoSQL. Hệ thống sử dụng để lưu log thao tác của người dùng, log giao tiếp với hệ thống HĐĐT |
| 10 | **IDP** | Là hệ thống đăng nhập tập trung (SSO) cung cấp các giao thức xác thực cho toàn bộ hệ thống tích hợp với VNPT Oneplace |
| 11 | **Logging** | Là hệ thống lưu trữ log tập trung được cung cấp mới IDG. Đây là nơi lưu trữ toàn bộ log của ứng dụng, sử dụng để kiểm tra, check lỗi. |

### 2.1. Hệ thống SSO

SSO của hệ thống là nền tảng được cung cấp bởi hệ thống IDP được phát triển bởi VNPT IT. Hệ thống được phát triển dựa trên nền tảng Wso2 Identity cung cấp ra nhiều phương thức xác thực khác nhau, với hệ thống HDDT sử dụng giao thức OIDC. Cơ chế xác thực SSO như hình dưới đây:

A diagram of a computer network

Description automatically generated

Hệ thống thực hiện SSO theo các bước như sau:

* 1: Người dùng truy cập web, ứng dụng, hoặc gọi tới api của hệ thống
* 2: Web app, Mobile app thực hiện redirect tới trang đăng nhập của SSO
* 3: Người dùng tiến hành đăng nhập và lấy về access token, id token
* 4: Client thực hiện gọi các api kèm theo access token để access vào hệ thống
* 5: Resouce server kiểm tra access token có hợp lệ hay không.

### 2.2. Caching

Hệ thống sử dụng Redis làm công nghệ để caching

Mô hình caching

Diagram

Description automatically generated

Giải thích

1. Hệ thống sử dụng Redis để thực hiện caching giữ liệu
2. Service thực hiện quản lý cache (Thêm, sửa, xóa cache) được thiết kế và chạy độc lập không ảnh hưởng tới các service nghiệp vụ.
3. Các service khi thay đổi dữ liệu cần cache. Thực hiện gửi bản tin giữ liệu cần cache lên Message broker, service cache sẽ nhận bản tin và thực hiện việc cache dữ liệu.
4. Giữ liệu cache được lưu trong redis với thời gian xác định nếu không được sử dụng. Sau thời gian này giữ liệu cache sẽ được xóa.
5. Các tính năng sử dụng tới cache, service thực hiện kiểm tra trong Redis nếu có dữ liệu thì đọc lên, nếu không có thì thực hiện đọc giữ liệu từ database và insert vào cache.

### 2.3 Logging

Diagram

Description automatically generated

Mô tả:

1. Hệ thống microservice sử dụng mô hình log tập trung, hạ tầng lưu trữ log, check log sử dụng hệ thống Elasticsearch và Kibana của IDG.
2. Các microservice sử dụng thư viện và gửi log về container Fluentd để thực hiện ghi log tập trung.
3. Các instance của cùng một service thực hiện gửi chung một Index log để dễ dàng trong việc kiểm tra log của service.
4. Để đảm bảo trace log trong distributed transaction. Tất cả các log được ghi xuống đều phải có transacetion Id.

### 2.4 Microservice communication

Hiện tại với hệ thống microservice có các phương pháp giao tiếp như sau:

1. Giao tiếp đồng bộ: Sử dụng Http, GRPC. Với các hệ thống phát triển mới ưu tiên sử dụng GRPC có tốc độ kết nối và xử lý giữ liệu nhanh hơn. Nhược điểm của 2 giao thức này là các service sẽ bị phụ thuộc chặt chẽ vào nhau, khi service được gọi có vấn đề thì sẽ gây lỗi cho các service gọi tới.
2. Giao tiếp bất đồng bộ: Sử dụng các message broker như RabbitMQ, Kafka, … Với giao tiếp bất đồng bộ thì các service sẽ ít phụ thuộc nhau hơn. Tất cả giao tiếp của hệ thống đều thông qua cơ chế Event và Handle. Ngược lại nhược điểm là quá trình implement chức năng sẽ tốn nhiều nguồn lực cũng như khi xảy ra lỗi thì quá trình debug lỗi sẽ khó khăn hơn nhiều.
3. Giao tiếp sử dụng service mesh của k8s.

Dưới đây là hình vẽ mô phỏng kết nối bất đồng bộ giữa các service trong hệ thống sử dụng message broker:

Diagram

Description automatically generated

### 2.5 Distributed Transaction

Một trong những vấn đề khó khăn nhất trong việc implement hệ thống microservice là giải quyết vấn đề Transaction. Khi mà một giao dịch của người dùng được phân tán và xử lý trên nhiều service vấn đề xảy ra làm sao đảm bảo được data consistency.

Để giải quyết vấn đề Distributed Transaction, chúng ta áp dụng Saga design pattent. Dưới đây là hình vẽ mô ta quá trình xử lý.

Luồng transaction thành công:

A diagram of a company

Description automatically generated

Luồng thất bại và rollback

Diagram

Description automatically generated

## 3. Kiến trúc triển khai

* Hệ thống được triển khai theo kiến trúc Microservices. Các service được đóng thành images và triển khai chạy trên nền tảng k8s dùng chung của công ty.
* Các service được deploy vào tối thiểu 2 pods trên nền tảng k8s để đảm bảo HA cho từng service. K8s có cơ chế helthcheck để kiểm tra service còn hoạt động hay không và restart service để đảm bảo service luôn chạy.
* Hệ thống được triển khai dạng Microservices trên nền tảng k8s có thể tự động giãn nở theo yêu cầu, các chức năng được tách biệt thành các service nhỏ và có thể được scale tự động theo tài nguyên cần để thực hiện.
* Dữ upload từ file người dùng, các file hệ thống cần lưu lại được lưu trữ trên hệ thống MinIO của IDG đảm bảo về an toàn dữ liệu, sao lưu backup định kỳ.
* Toàn bộ các database bao gồm mysql và mongo database được triển khai Master-Slave và nằm ngoài cụm K8S. Dữ liệu database được backup theo ngày, lưu trữ trong vòng 90 ngày và có thể khôi phục lại trong vòng 2h nếu xảy ra lỗi.

A diagram of a flowchart

Description automatically generated

## 4. Kiến trúc Multi-tenancy (SaaS)

Mô tả:

* Dữ liệu của các đơn vị được lưu chung trong cùng table (collection), cùng database đối với cả trên MySQL lẫn MongoDB
* Các microservices sẽ sử dụng các schema riêng trên MySQL và collection riêng ở trên MongoDB.
* Dữ liệu của các đơn vị được phân biệt với nhau bằng enterprise\_id:
* Trong các câu SQL thao tác với dữ liệu luôn được truyền theo enterprise để đảm bảo chỉ thao tác với dữ liệu của đơn vị cần cập nhật.
* Khi thực hiện cập nhật trên production chỉ có người quản trị được phép cập nhật và phải luôn tuân thủ theo policy
* Các dữ liệu sinh ra nhiều như: Log thao tác người dùng, các dữ liệu về log giao dịch giữa các hệ thống được lưu trữ trên MongoDB với khả năng scale và xử lý dữ liệu tốc độ cao mạnh mẽ.
* Ngoài database quản trị là dùng chung với lượng giữ liệu quản trị người dùng không quá lớn. Các database giữ liệu hóa đơn được tách ra theo tỉnh thành phố. Nên dữ liệu của từng database sẽ không quá lớn mức Table.

# III. TIÊU CHUẨN CHẤT LƯỢNG

Hệ thống được thiết kế kiến trúc như trên sẽ đảm bảo các tiêu chuẩn chất lượng như sau:

## Hiệu năng

|  |  |  |
| --- | --- | --- |
| Mã | Yêu cầu | Đặc tả |
| 1 | Yêu cầu về thời gian | Các yêu cầu về thời gian phổ biến như thời gian phản hồi, thời gian xử lý và thời gian trả kết quả khi một hệ thống thực hiện các chức năng đáp ứng yêu cầu người sử dụng trong trạng thái hoạt động bình thường của hệ thống. Yêu cầu về thời gian cần được lượng hóa đảm bảo khả năng đánh giá được. Đảm bảo thời gian cho phép để hệ thống phản hồi lại thông tin đã tiếp nhận yêu cầu xử lý từ phía người sử dụng.  Thời gian tối đa để thực hiện xử lý cập nhật không quá 10s, xử lý thống kê không quá 60s. |
| 2 | Hiệu năng | Hệ thống thiết kế đáp ứng 200CCU và có khả năng mở mở rộng khi có yêu cầu.  ***Ghi chú***: *Hiệu năng đáp ứng với các thành phần do SI xây dựng, không áp dụng cho các hệ thống ngoài như: HDDT, Smart CA, VNPT Email, Workplace.* |

## Tính khả dụng

|  |  |  |
| --- | --- | --- |
| Mã | Yêu cầu | Đặc tả |
| 1 | Phù hợp với nhu cầu | Hệ thống phù hợp với nhu cầu người sử dụng |
| 2 | Khả năng giúp người sử dụng học được | Khả năng hệ thống hỗ trợ khả năng học cách sử dụng đối với nhóm các đối tượng sử dụng khác nhau nhằm thực hiện được các tác vụ trong một ngữ cảnh cụ thể, kết quả đáp ứng được yêu cầu của đối tượng sử dụng. |
| 3 | Giao diện người sử dụng | Có khả năng hiển thị, hoạt động chính xác, đầy đủ trên hầu hết các trình duyệt phổ biến phiên bản mới nhất (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari,...). |
| 4 | Khả năng truy cập | Hệ thống dễ truy cập từ môi trường Internet |

* Tính sẳn sàng của hệ thống tính theo phần trăm: 99 %
* Hệ thống chỉ downtime vào những lúc nào: khi nâng cấp hệ thống, nâng cấp tối đa 2 lần 1 ngày vào lúc 12h -> 13h và 23h -> 24h. Mỗi lần nâng cấp thời gian tối đa 5ph.

## Tính an toàn – bảo mật

|  |  |  |
| --- | --- | --- |
| Mã YC | Yêu cầu | Đặc tả |
|  | **Thông tin định danh** |  |
| 1 | Tên đăng nhập | Tên đăng nhập là email của người dùng và phải là duy nhất. |
| 2 | Thiết lập chính sách mật khẩu | Thiết lập chính sách mật khẩu đảm bảo:   * Mật khẩu phải tối thiểu 8 ký tự, có đầy đủ chữ hoa, chữ thường, số và ký tự đặc biệt. * Đăng nhập sai quá 5 lần tiến hành khóa tài khoản 1 tiếng. |
| 3 | Chức năng Reset/ quên mật khẩu | Đối với chức năng Reset/ quên mật khẩu:   * Chương trình tiến hành gửi đường link dẫn tới trang quên mật khẩu kèm theo 1 token sử dụng 1 lần tới email của người dùng. Người dùng tiến hành reset mật khẩu 1 lần duy nhất ứng với token được gửi. |
| 4 | Mã Hash của mật khẩu trong DB | Mật khẩu được lưu trữ trên DB với mã HASH 256 bits. Được mã hóa 1 chiều và không thể giải mã. |
| 5 | **Xử lý xác thực** | 1. Không trả về thông báo chi tiết thông tin định danh cho trường hợp người dùng đăng ký thông tin định danh (Username, Email,...) đã tồn tại tại chức năng đăng ký, hoặc gửi sai thông tin định danh tại các chức năng đăng nhập, Reset/quên mật khẩu, đổi địa chỉ Email. 2. Chỉ sử dụng phương thức POST để Submit thông tin định danh, sử dụng HTTPS cho mọi giao thức. |
| 6 | **Quản lý phiên đăng nhập** | Session phải được sinh ngẫu nhiên và độ dài tối thiểu là 128-bit.  Session có timeout là 2 giờ. Trong vòng 2 giờ nếu người dùng không có thao tác gì với hệ thống sẽ tự động đăng xuất. Thời gian sống tối đã của 1 session sẽ không quá 1 ngày. Sau thời gian đó người dùng sẽ phải đăng nhập lại.  Tạo mới Session sau khi đăng nhập thành công.  Xóa giá trị Session ID và các dữ liệu gắn với Session đó khi người dùng đăng xuất.  Cấu hình thuộc tính “Secure” đối với các ứng dụng sử dụng HTTPS và “HTTP-Only” cho trường Session Cookie. |
| 7 | **Phân quyền** | Hệ thống được phân quyền theo role của người dùng: Mỗi người dùng được gán vào 1 hoặc nhiều role cố định của hệ thống. Mỗi role được gán cho các quyền truy cập, gọi API của các màn hình chức năng khác nhau. Khi đăng nhập role của người dùng được đưa vào trong token của người dùng, session cookies và được kiểm tra mỗi khi gọi màn hình hoặc gọi API để cho phép truy cập hay không. |
| 8 | **Mã hóa các dữ liệu nhạy cảm** | Mã hóa các dữ liệu nhạy cảm trước khi lưu trữ (password, ...) |
| 9 | **SQL** | Các thao tác xuống Database đều được xác thực mỗi lần mở connection. Các thao tác đều được gọi qua Procedures và truyền parameter để chống SQL Injection. |
| 10 | **Tương tác với File** | 1. Không truyền trực tiếp dữ liệu từ người dùng đến các hàm include File. 2. Lập danh sách các định dạng file được phép upload. Chỉ cho phép 1 số loại file cơ bản được upload như word, excel,… không cho phép upload file .exe, .bat và giới hạn kích thước của file. 3. Validate file hợp lệ bằng cách kiểm tra đồng thời File header và phần mở rộng của File. 4. Với các trường hợp không bắt buộc thì không lưu File upload trong thư mục Web, bỏ quyền thực thi trên thư mục Upload. 5. Khi cần tham chiếu tới các File tồn tại trên hệ thống cần thiết lập danh sách đầu vào mong muốn hoặc gán các giá trị định danh tương ứng cho các File thay vì truyền tên File. 6. Không trả về đường dẫn tuyệt đối của File được lưu trữ trong sản phẩm, dịch vụ. 7. Tất cả dữ liệu, tài nguyên hệ thống (Báo cáo, File upload, File cấu hình...) không được lưu trong thư mục cho phép truy cập trực tiếp không qua xác thực. |
| 14 | **Xử lý Back-End HTTP Request** | Khi tạo HTTP request phía server, các tham số GET, POST cho yêu cầu đó tránh tạo từ dữ liệu phía người dùng, hoặc phải được kiểm tra cẩn thận để chống ghi đè các tham số khác. Tất cả các request đều chạy trên nền HTTPS không sử dụng giao thức HTTP. |
| 15 | **Kiểm soát dữ liệu đầu vào** | 1. Việc kiểm tra dữ liệu đầu vào phải được thực hiện phía dịch vụ. Không trực tiếp xử lý dữ liệu do người dùng trực tiếp nhập vào, yêu cầu phải kiểm tra có hợp lệ hay không và thông báo cho người dùng. 2. Thực hiện việc kiểm tra dữ liệu từ tất cả các nguồn dữ liệu có tương tác với người dùng (Các tham số lấy từ GET/POST request, HTTP Headers, dữ liệu lấy từ DB, dữ liệu từ File upload,...). 3. Xác định rõ chuẩn định dạng encode của dữ liệu đầu vào, thực hiện validate dữ liệu sau khi đã Decode đầu vào về 1 định dạng chuẩn và nhất quán. 4. Validate kiểu dữ liệu, phạm vi và độ dài dữ liệu và định dạng dữ liệu. 5. Nếu dữ liệu đầu vào bắt buộc là các ký tự đặc biệt, cần thiếp lập danh sách các ký tự đầu vào mong muốn. Lọc dữ liệu đầu vào người dùng, loại bỏ các ký tự độc hại phổ biến <, > , ‘ , « , %, &, +, or, \\, \ … để chống các hình thức SQL injection, XSS |
| 16 | **Kiểm soát dữ liệu đầu ra** | 1. Phải chỉ rõ Character Encoding cho dữ liệu đầu ra. 2. Response body phải được encode theo ngữ cảnh sử dụng. Một số trường hợp phổ biến:  * Đầu ra là HTML, thực hiện HTML Encode các kí tự đặc biệt (<>’”&) từ các nguồn dữ liệu không an toàn (Các tham số lấy từ GET/POST request, HTTP Headers, dữ liệu lấy từ DB, dữ liệu từ File upload,...có thể điều khiển được bởi người dùng). * Đầu ra là Json, thực hiện Encode dữ liệu trả về dạng Object, không trả về dạng mảng.  1. Response header: lọc bỏ các kí tự đặc biệt (\n, \r) do dữ liệu người dùng truyền vào. 2. Cookie trả về cần giới hạn tối thiểu nhất các thuộc tính (Domain, Path, HTTP Only, Expire, Secure). Tránh lưu trữ các dữ liệu nhạy cảm trên Cookie, nếu cần lưu trữ các dữ liệu nhạy cảm thì phải thực hiện mã hóa các dữ liệu này với thuật toán đối xứng mạnh và key chỉ được lưu trên Server. 3. Hạn chế việc chuyển hướng, chuyển tiếp đến các URI khác. Nếu ứng dụng có chức năng này cần phải lập danh sách các URI được phép thực hiện chuyển hướng, chuyển tiếp. |
| 17 | **Kiểm soát ngoại lệ và ghi log ứng dụng.** | Xử lý các ngoại lệ bằng Try-Catch và trả về các thông báo lỗi chung đã Custom (không show exception ra giao diện chương trình), thông báo lỗi trả về không được chứa các thông tin nhạy cảm của người dùng, hệ thống,...  Các thông tin lỗi, ngoại lệ này phải được log lại để phục vụ bảo trì, xác định nguyên nhân lỗi ứng dụng.  File log không được đặt cùng thư mục mã nguồn của hệ thống.  Không lưu thông tin nhạy cảm (như thông tin mật khẩu, Session ID,…) trong file log.  Giới hạn người dùng cho phép truy cập file log. |
| 18 | **Sử dụng Framewwork, Lib (thành phần bên thức ba)** | Sử dụng framework .NET 8 của Microsoft và thư viện chính thức Devexpress, hệ thống chạy trên nền tảng K8S. |
| 19 | **Các yêu cầu khác** | Theo Quyết định số 854/QĐ-VNPT-IT-CLGSP-ĐHIT ngày 30/10/2018 về việc ban hành quy định an toàn,an ninh sản phẩm phần mềm trước khi cung cấp dịch vụ, 89/QĐ-VNPT-IT&VAS ngày 25/01/2018 Quy định đảm bảo an toàn hệ thống thông tin trong hoạt động của VNPT |

## Tính sửa chữa và bảo trì

Hệ thống được thiết kế trên nền tảng Microservices. Các thành phần hệ thống tách biệt rõ ràng: Web client, API Gateway, Invoice API, Tax API, Background Service, IdentityAPI, … Do đó khi nâng cấp hoặc cập nhật các thành phần riêng rẽ sẽ không ảnh hưởng tới các thành phần khác. Hệ thống được triển khai trên nền tảng K8S dùng chung của IDG đảm bảo tính HA cao.

Database dữ liệu của khách hàng được chia nhỏ ra nhiều schema. Nên khi nâng cấp cũng như bảo trì sẽ gây ảnh hưởng tới một số nhóm khách hàng nhất định chứ không ảnh hưởng toàn hệ thống.

## Tính tự kiểm thử

Mức độ các module, chức năng có thể áp dụng việc kiểm thử tự động: hệ thống chưa áp dụng test tự động.

## Tính mở rộng

* Hệ thống được thiết kế trên mô hình microservices chạy trên nền tảng K8S. Tất cả các thành phần của hệ thống có thể co giãn theo chiều ngang. Khi nhu cầu sử dụng cả các thành phần tăng lên có thể scale thành phần đó thành nhiều instance để chạy và cũng có thể giảm thành phần đó nếu nhu cầu sử dụng ít đi. Cụm server K8S có thể được mở rộng một cách đơn giản bằng cách bổ sung các worker hoặc master node.
* Hệ thống thực hiện co dãn một cách tự động thông qua cấu hình HPA trên nền tảng K8S.
* Thời gian thực hiện co giãn của các service trong hệ thống trong vòng 5s.

## Tính tái sử dụng

* Toàn bộ các thành phần trong hệ thống được xây dựng theo dạng Class Library đóng thành các thư viện riêng rẽ để sử dụng. Các hàm và API đều được viết theo chuẩn chung HTTP rõ ràng có thể được tái sử dụng cho các hệ thống khác nếu có nhu cầu.

## Tính hỗ trợ

* Khi hệ thống có lỗi (unhandled exception), lập trình viên, quản trị viên kiểm tra log của hệ thống ở file log của ứng dụng hoặc sử dụng hệ thống log của K8S để xác định nguyên nhân lỗi.
* Khi có sự cố downtime, quản trị viên kiểm tra log của hệ thống K8S để xác định nguyên nhân và khắc phục.
* Quản trị viên theo dõi các thông số của hệ thống CPU, RAM thông qua phần mềm MoBaExtern.

## Tính tích hợp

* Hiện tại hệ thống đang cung cấp ra các API trên nền giao thức HTTPS để các hệ thống khác tích hợp vào (ĐHSX, OneSME). Ngoài ra hệ thống cũng có thể tích hợp với hệ thống khác qua giao thức Message Broker (Kafka, RabbitMQ)
* Tất cả các giao thức tích hợp đều được xác thực rõ ràng thông qua UserName, Password, Token

## Tính khôi phục thảm họa

* Hệ thống được triển khai trên trung tâm dữ liệu của VNPT, được sao lưu backup và server được đặt trên nhiều vùng miền của đất nước, có khả năng khôi phục cao lúc xảy ra thảm họa.

# PHỤ LỤC I: CÁC YẾU TỐ ẢNH HƯỞNG

Kiến trúc phần mềm được thiết kế ở trên bị ảnh hưởng bởi các yếu tố sau:

## Yếu tố về thời gian và ngân sách

* Thời gian đảm bao đúng tiến độ bàn giao của dự án , theo từng giai đoạn
* Ngân sách đảm bao theo đúng ngân sách của dự án.

## Yếu tố về công nghệ

|  |  |  |
| --- | --- | --- |
| **Công nghệ sử dụng** | **Loại** | **Mô tả** |
| MySql 8.0.15 | Hệ quản trị CSDL | Cần sử dụng ít nhất 2 server CSDL, cài đặt cơ chế động bộ Master Slave. MySql with InnoDB support |
| NginX | Web Server |  |
| C#  JavaScript | Ngôn ngữ lập trình | C#11 |
| .NET 8 | Framework | Sử dụng .NET8 framework mã nguồn mở mới nhất của Microsoft |
| Devexpress | Library | Sử dụng cho các component của giao diện, Report |
| K8S | Nền tảng triển khai ứng dụng | Là nền tảng để chạy các service của dịch vụ trên nền container. |

## Yếu tố về nhân sự

Bản thiết kế kiến trúc phần mềm này không bị ảnh hưởng bởi số lượng nhân sự dự án hiện tại