BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

****

**BÁO CÁO THỰC TẬP DOANH NGHIỆP**

**CÔNG TY CỔ PHẦN TVT GROUP VIETNAM**

**Back-End Developer**

**Chuyên ngành đào tạo:** Hệ thống thông tin

**Giảng viên giám sát:** ThS. Huỳnh Nam

**Người hướng dẫn thực tập:** Nguyễn Ngọc Tú

**Họ và tên sinh viên thực tập:** Lê Văn Việt Anh

**Mã sinh viên:** 21101911

**Lớp:** DHHTTT17BTT

**Thành phố Hồ Chí Minh – Tháng 11 Năm 2025**

BỘ CÔNG THƯƠNG

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

****

**BÁO CÁO THỰC TẬP DOANH NGHIỆP**

**CÔNG TY CỔ PHẦN TVT GROUP VIETNAM**

**Back-End Developer**

**Chuyên ngành đào tạo:** Hệ thống thông tin

**Giảng viên giám sát:** ThS. Huỳnh Nam

**Người hướng dẫn thực tập:** Nguyễn Ngọc Tú

**Họ và tên sinh viên thực tập:** Lê Văn Việt Anh

**Mã sinh viên:** 21101911

**Lớp:** DHHTTT17BTT

**Thành phố Hồ Chí Minh – Tháng 11 Năm 2025**

**LỜI CẢM ƠN**

Lời đầu tiên, em xin bày tỏ lòng biết ơn chân thành và sâu sắc nhất đến Ban Giám Hiệu cùng toàn thể quý Thầy Cô trường Đại học Công nghiệp Thành phố Hồ Chí Minh. Quý Thầy Cô đã luôn tận tâm chỉ bảo, truyền đạt những kiến thức quý báu, tạo nền tảng vững chắc và hành trang cần thiết để chúng em tự tin bước vào tương lai.

Em cũng xin gửi lời cảm ơn đặc biệt đến Ban Lãnh đạo CÔNG TY CỔ PHẦN TVT GROUP VIETNAM cùng toàn thể các anh chị Back-end Developer trong Đội ngũ Back-end. Các anh chị đã tạo mọi điều kiện thuận lợi, nhiệt tình hỗ trợ và dành thời gian quý báu để em được tiếp cận thực tế, trau dồi kiến thức, kỹ năng lập trình nghiệp vụ. Sự giúp đỡ này là yếu tố quan trọng nhất giúp em hoàn thành tốt bài báo cáo thực tập này.

Tuy nhiên, do kiến thức thực tế còn hạn chế, kinh nghiệm ứng dụng vào thực tiễn chưa nhiều, bài báo cáo thực tập của em chắc chắn không thể tránh khỏi những thiếu sót và hạn chế nhất định. Em kính mong nhận được những ý kiến đóng góp, chỉ dẫn quý báu từ quý Thầy Cô và các anh chị để em có thể hoàn thiện kiến thức và kỹ năng của bản thân.

Cuối cùng, em xin kính chúc quý Thầy Cô, Ban Lãnh đạo và toàn thể anh chị cán bộ nhân viên CÔNG TY CỔ PHẦN TVT GROUP VIETNAM luôn dồi dào sức khỏe, hạnh phúc và gặt hái nhiều thành công trong công việc.

**PHIẾU NHẬN XÉT CƠ QUAN THỰC TẬP**

TP.HCM, ngày …… tháng …… năm……

**XÁC NHẬN CỦA CƠ QUAN**

**ĐÁNH GIÁ KẾT QUẢ THỰC TẬP CỦA GIẢNG VIÊN GIÁM SÁT**

TP.HCM, ngày …… tháng …… năm……

**GIẢNG VIÊN GIÁM SÁT**

**LỜI NÓI ĐẦU**

Thế giới đang chứng kiến những bước tiến vượt bậc của cuộc Cách mạng Công nghiệp lần thứ tư, trong đó, **Công nghệ Thông tin (CNTT) và Phát triển Phần mềm** được xem là hạt nhân, quyết định năng lực cạnh tranh và tốc độ phát triển của mỗi quốc gia. Ngành CNTT không chỉ cung cấp các giải pháp số hóa cho mọi lĩnh vực kinh tế mà còn tiên phong trong việc kiến tạo các công nghệ đột phá như Trí tuệ Nhân tạo (AI), Internet Vạn vật (IoT), và các ứng dụng di động thông minh. Việc làm chủ và phát triển các công nghệ cao như **Xử lý hình ảnh (Image Processing)** hay **Công nghệ tương tác thực tế ảo (Augmented Reality)** đang trở thành yêu cầu thiết yếu để tạo ra sự khác biệt và giá trị gia tăng trên thị trường toàn cầu.

Trong dòng chảy mạnh mẽ đó, **CÔNG TY CỔ PHẦN TVT GROUP VIETNAM** đã khẳng định vị thế của mình là một đơn vị tiên phong, hoạt động dựa trên nguyên tắc **lựa chọn và tập trung** vào các giải pháp công nghệ kỹ thuật cao. Tiền thân là công ty **Covisoft** – một đơn vị dày dặn kinh nghiệm trong nghiên cứu và phát triển ứng dụng hệ thống định vị, chỉ dẫn (Navigation) tại Hàn Quốc – TVT GROUP được thành lập tại Việt Nam từ tháng 4/2012. Trải qua quá trình hình thành và phát triển (chính thức đổi tên thành TVT GROUP vào tháng 08/2019), Công ty đã mở rộng phạm vi hoạt động, không chỉ duy trì thế mạnh cốt lõi mà còn tập trung mạnh mẽ vào lập trình phần mềm ứng dụng đa nền tảng cho các thiết bị di động thông minh (Android, iOS, Window Phone, v.v.).

**Tầm nhìn** của TVT GROUP là trở thành đơn vị hàng đầu trong việc nghiên cứu và phát triển những phần mềm ứng dụng mang tính kỹ thuật cao, phù hợp với xu thế chung của nền công nghệ thế giới, nhằm khẳng định chất lượng sản phẩm và thương hiệu tại thị trường Châu Á. Để hiện thực hóa tầm nhìn này, Công ty đã xây dựng một triết lý hoạt động sâu sắc, đặt nền tảng trên **Triết lý 5C** (**Challenge – Thách thức, Collaboration – Hợp tác, Customer Focus – Định hướng Khách hàng, Confidence – Tự tin, Change – Thay đổi**) cùng với các giá trị văn hóa doanh nghiệp cốt lõi:

* **Tinh thần Đội nhóm và Hợp tác:** Công ty luôn tạo điều kiện để các cá nhân làm việc theo đội, tin rằng một tập thể hợp tác và hiệu quả là yếu tố thiết yếu cho sự thành công. Sự đóng góp và đồng lòng của từng thành viên chính là sức mạnh của cả Công ty.
* **Minh bạch và Trao đổi Thông tin:** Nhân viên được khuyến khích chia sẻ quan điểm, tìm kiếm thông tin và giải quyết những vướng mắc cùng với cấp trên trực tiếp, đảm bảo mọi thành viên được lắng nghe và mọi vấn đề được giải quyết một cách nhanh chóng, hiệu quả nhất.
* **Tôn trọng và Phát triển Nhân viên:** TVT GROUP xem con người là nguồn lực quý giá nhất, luôn tạo mọi điều kiện để nhân viên phát huy hết tiềm năng, khuyến khích học hỏi và tin tưởng giao phó những trọng trách cao hơn, phù hợp với khả năng và kinh nghiệm.
* **Trách nhiệm và Chuyên nghiệp:** Sự thành công của Công ty gắn liền với việc thỏa mãn quyền lợi hợp lý của khách hàng và nhân viên. Điều này đòi hỏi mọi thành viên phải làm việc đúng giờ, hoàn thành công việc hữu hiệu, đạt chất lượng cao, đồng thời thể hiện **Tính chuyên nghiệp** qua cam kết cung cấp giải pháp chất lượng, dịch vụ hậu mãi chu đáo và giao tiếp lịch thiệp, tôn trọng.
* **Đóng góp Cộng đồng:** Ban Lãnh đạo và toàn thể nhân viên Công ty tự nguyện tham gia các hoạt động từ thiện hoặc các công tác xã hội để phát triển cộng đồng, thể hiện trách nhiệm xã hội và tuân thủ pháp luật nghiêm túc.

Nhận thấy môi trường lý tưởng này, đặc biệt là cơ hội được tham gia vào các dự án phát triển công nghệ cao và rèn luyện kiến thức chuyên môn **Back-end Developer**, em đã lựa chọn **CÔNG TY CỔ PHẦN TVT GROUP VIETNAM** là nơi thực tập. Giai đoạn thực tập không chỉ là cơ hội để áp dụng kiến thức lý thuyết đã học tại **Đại học Công nghiệp Thành phố Hồ Chí Minh** vào thực tiễn mà còn là thời gian quý báu để trau dồi kỹ năng mềm và tác phong làm việc chuyên nghiệp. Sau hai tháng thực tập nỗ lực, em xin trình bày báo cáo tổng hợp này, ghi lại quá trình trải nghiệm và những kiến thức đã thu nhận được.

Nội dung báo cáo gồm 2 chương:

Chương 1: Giới thiệu chung về CÔNG TY CỔ PHẦN TVT GROUP VIETNAM

Chương 2: Nội dung thực tập tại CÔNG TY CỔ PHẦN TVT GROUP VIETNAM

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ NGÂN HÀNG TMCP XUẤT NHẬP KHẨU 20](#_Toc215727651)

[1.1. **Thông tin chung về đơn vị thực tập** 20](#_Toc215727652)

[1.2. Chức năng và Tầm nhìn 21](#_Toc215727653)

[1.2.1. Chức năng của công ty 21](#_Toc215727654)

[1.2.2. Tầm nhìn của công ty 21](#_Toc215727655)

[1.3. Quá trình hình thành và chính sách nhân sự của công ty 24](#_Toc215727656)

[1.4. Quy trình làm việc & Quy định làm việc 25](#_Toc215727657)

[1.4.1. Quy trình làm việc 25](#_Toc215727658)

[1.4.2. Quy định làm việc 34](#_Toc215727659)

[1.5. Quy trình kiểm thử của team QC 36](#_Toc215727660)

[1.6. **Quy trình test phần mềm trong quá trình sản xuất phần mềm.** 43](#_Toc215727661)

[CHƯƠNG 2: NỘI DUNG THỰC TẬP TẠI CÔNG TY CỔ PHẦN TVT GROUP 44](#_Toc215727662)

[2.1. Giới thiệu về luồng dữ liệu: 44](#_Toc215727663)

[2.2. Công cụ hỗ trợ phát triển – MedusaJS Framework 45](#_Toc215727664)

[2.3. Công cụ và nền tảng xác thực – Kinde Authentication 45](#_Toc215727665)

[2.4. Thực tế quy trình tiếp nhận và xử lý yêu cầu lấy dữ liệu tại Eximbank 46](#_Toc215727666)

[2.4.1. Khởi tạo và gửi yêu cầu 46](#_Toc215727667)

[2.4.2. Tiếp nhận và thẩm định yêu cầu kỹ thuật 46](#_Toc215727668)

[2.4.3. Triển khai, trích xuất và xử lý dữ liệu 47](#_Toc215727669)

[2.4.4. Bàn giao kết quả và đóng yêu cầu 47](#_Toc215727670)

[2.5. Thực tế quy trình tiếp nhận và xử lý yêu cầu lấy dữ liệu tại Eximbank 47](#_Toc215727671)

[KẾT LUẬN 48](#_Toc215727672)

# 

# DANH MỤC HÌNH ẢNH

[Hình 1.1. Logo CÔNG TY CỔ PHẦN TVT GROUP VIETNAM 21](#_Toc215572118)

[Hình 1.2. Quy trình làm việc trong dự án 41](#_Toc215572119)

[Hình 1.3. Types of tools 21](#_Toc215572118)

# 

# Thời gian làm việc và nội dung công việc ở công ty

**Thời gian làm việc:**  
 Công ty áp dụng khung thời gian làm việc chuẩn, đồng thời cho phép linh hoạt theo đặc thù dự án. Cụ thể:

* Ngày làm việc: Thứ Hai đến Thứ Sáu
* Giờ làm việc: 09:00 – 18:00
* Thời gian nghỉ trưa: 11:50 – 13:00
* Ngày nghỉ hằng tuần: Thứ Bảy và Chủ Nhật

Tùy theo yêu cầu của dự án hoặc khách hàng, nhân viên có thể chủ động điều chỉnh thời gian làm việc trong ngày, nhưng vẫn phải đảm bảo tổng thời gian tối thiểu 8 giờ mỗi ngày và 40 giờ mỗi tuần. Mỗi bộ phận tự quản lý tiến độ và thời gian để bảo đảm hoàn thành nhiệm vụ đúng hạn.

**Nội dung công việc ở công ty:**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Thời gian | Nội dung công việc | Kết quả | Hình ảnh |
| **Tuần thứ 1**  **30/09/2025** | Làm quen với nội quy, văn hóa làm việc và quy trình của doanh nghiệp. | Đã nắm được nội quy, cách thức giao tiếp và làm việc trong môi trường công ty. |  |
| **01–02/10/2025** | Bắt đầu tìm hiểu cấu trúc MedusaJS framework: các module, cấu trúc thư mục, và cơ chế hoạt động. | Hiểu được kiến trúc tổng thể và vai trò của từng thành phần trong framework. |  |
| **03/10/2025** | Thực hành khởi chạy dự án mẫu bằng MedusaJS framework. | Đã khởi chạy thành công project mẫu, hiểu được luồng hoạt động cơ bản |  |
| **04/10/2025** | Tìm hiểu thêm tài liệu hướng dẫn và cộng đồng hỗ trợ (MedusaJS Docs, GitHub). | Gặp khó khăn trong việc tìm tài liệu do framework còn mới, nhưng đã dần cải thiện kỹ năng research. |  |
| **Tuần thứ 2**  **7/10/2025-8/10/2025** | Nghiên cứu cách xây dựng **Custom Features** trong MedusaJS. | Hiểu cách framework hỗ trợ mở rộng tính năng thông qua Module, Workflow và API Route |  |
| **08–09/10/2025** | Xây dựng tính năng **Thương Hiệu sản phẩm (Brand)** cho hệ thống quản trị. | Hoàn thành API thêm thương hiệu, tạo Module và Workflow cho tính năng “Add Brand”. |  |
| **10/10/2025** | Tùy chỉnh **Admin Dashboard** để hiển thị và quản lý thương hiệu. | Giao diện quản trị hiển thị được danh sách thương hiệu; thêm chức năng chỉnh sửa, xóa. |  |
| **11/10/2025** | Tích hợp hệ thống bên thứ ba (**CMS**) để đồng bộ dữ liệu thương hiệu và nội dung. | Hoàn thành việc tích hợp CMS, đảm bảo đồng bộ thương hiệu và thiết lập lịch trình đồng bộ hàng ngày. |  |
| **Tuần thứ 3**  **13/10/2025** | Chuẩn bị slide báo cáo tiến độ nghiên cứu framework. | Hoàn thiện bản nháp slide, sẵn sàng cho buổi báo cáo. |  |
| **14/10/2025** | Tìm hiểu tổng quan (overview) về MedusaJS framework. Đọc tài liệu hướng dẫn và phân tích kiến trúc hệ thống. | Hiểu rõ cấu trúc tổng quan và luồng hoạt động của framework. |  |
| **15/10/2025** | Nghiên cứu chi tiết về Medusa Container và Modules. Tìm hiểu cơ chế dependency injection, service container và module loader. | Nắm được cách container quản lý modules và mối quan hệ giữa chúng. |  |
| **16/10/2025** | Tự học bổ sung về NodeJS và ExpressJS. Thực hành xây dựng ứng dụng nhỏ để hiểu rõ cấu trúc ExpressJS | Hiểu được sự tương đồng giữa ExpressJS và MedusaJS. |  |
| **17/10/2025** | Tổng hợp nội dung và xác định mục tiêu tuần tới.  Hoàn thiện slide, ghi chú kiến thức đã học, lập kế hoạch xây dựng module thử nghiệm. | Sẵn sàng cho buổi báo cáo tiến độ với người hướng dẫn. |  |
| **Tuần 4**  **20/10/2025** | Hoàn thành báo cáo tiến độ nghiên cứu về MedusaJS framework. | - Báo cáo tiến độ được hoàn thành và phản hồi tích cực. - Được góp ý nên đi sâu hơn vào phần triển khai thực tế chức năng trong framework. |  |
| **21/10/2025** | Đào sâu nghiên cứu Medusa Container và Modules.  Gặp khó khăn trong việc hiểu cú pháp và cách gọi hàm trong TypeScript/JavaScript. | - Hiểu cơ bản cơ chế **dependency injection** trong Medusa.  - Ghi nhận được những điểm cần củng cố về cú pháp code. |  |
| **22/10/2025** | Tự học ExpressJS framework để củng cố nền tảng NodeJS.  - Xây dựng ứng dụng web nhỏ sử dụng ExpressJS (tạo API GET/POST, routing).  - So sánh cấu trúc Express với Medusa để hiểu cách Medusa mở rộng ExpressJS. | - Nắm vững hơn về cú pháp JavaScript (async/await, import/export, middleware).  - Hiểu cách framework Medusa kế thừa và mở rộng từ ExpressJS. |  |
| **23/10/2025** | Ứng dụng kiến thức từ ExpressJS để quay lại nghiên cứu MedusaJS.  - Đọc lại phần container và module trong Medusa, thử chỉnh sửa và chạy thử một module đơn giản.  - Ghi chú lại quy trình nạp module và khởi tạo container. | - Giải quyết được phần lớn khó khăn về cú pháp.  - Hiểu rõ mối quan hệ giữa module, service, và container trong Medusa. |  |
| **24/10/2025** | Tổng kết tuần và lập kế hoạch nghiên cứu tiếp theo.  Xác định mục tiêu tuần tới: thử **xây dựng module thực tế** | - Hoàn thành báo cáo tuần.  - Có định hướng rõ ràng cho bước triển khai chức năng thử nghiệm. |  |
| **Tuần 5**  **27/10/2025** | Khởi động với việc thực hành xây dựng ứng dụng web nhỏ bằng ExpressJS.  Tạo cấu trúc project, chia router – controller – service.  Viết các API GET và POST theo chuẩn REST.  Ôn lại cú pháp JavaScript: export/import, require, async/await | Hiểu rõ cách tổ chức dự án ExpressJS.  Các API GET/POST chạy ổn định.  Nắm chắc hơn các cú pháp bất đồng bộ trong JavaScript. |  |
| **28/10/2025** | Tiếp tục hoàn thiện API PUT, PATCH và DELETE.  Thêm middleware xử lý lỗi và validate dữ liệu đầu vào.  Kiểm thử API bằng Postman. | Hoàn thiện toàn bộ CRUD cho ứng dụng ExpressJS.  Hiểu sâu hơn về middleware và flow xử lý request.  Giảm lỗi khi thao tác dữ liệu. |  |
| **29/10/2025** | Ứng dụng kiến thức ExpressJS để chuyển sang nghiên cứu MedusaJS.  Đọc tài liệu về Container, Service, Module trong MedusaJS.  Tìm hiểu quan hệ giữa module → service → container trong hệ thống. | Hiểu cách MedusaJS tổ chức module và dependency injection.  Nắm rõ luồng hoạt động nội bộ của MedusaJS  . |  |
| **30/10/2025** | Thử xây dựng module đơn giản “Blogs” theo tài liệu MedusaJS.  Khởi tạo module mới, tạo service và API handler.  Chạy thử và debug các lỗi phát sinh. | Module Blogs chạy được và đã hiển thị log trên server.  Hiểu rõ cách khởi tạo module và đăng ký vào container. |  |
| **31/10/2025** | Tổng kết tuần.  Lập kế hoạch cho tuần sau: Xác định mục tiêu nghiên cứu về Auth Module trong tuần 6. | Module Blogs hoạt động ổn định.  Hoàn thành toàn bộ mục tiêu tuần 5. |  |
| **Tuần 6**  **03/11/2025** | Đọc tổng quan kiến trúc Auth Module.  Nghiên cứu sự khác nhau giữa scope (admin/store) và provider.  Tìm hiểu luồng đăng nhập → xác thực → trả về token. | Hiểu rõ sơ đồ hoạt động: Client → Auth Module → Provider → Token.  Nắm rõ cách tổ chức scope để phân quyền. |  |
| **04/11/2025** | Cấu hình Auth Module trong file medusa-config.ts.  Khai báo provider mặc định như emailpass.  Tìm hiểu JWT và cách Medusa quản lý session. | Auth Module load thành công khi khởi động server.  medusa-config.ts đã được cập nhật đầy đủ options.  Hiểu payload của JWT trong Medusa. |  |
| **05/11/2025** | Phân tích interface AbstractAuthModuleProvider.  Tìm hiểu các phương thức: authenticate, validate, verify.  Phân tích structure AuthenticationResponse để nắm input/output. | Nắm được cách custom provider.  Hiểu rõ yêu cầu của các hàm bắt buộc trong Provider. |  |
| **06/11/2025** | Thực hành tạo Actor type mới (Vendor/Shipper).  Định nghĩa data model.  Chạy generate migration để tạo bảng trong database.  Đăng ký Actor mới vào Auth Module, gán scope riêng. | Database đã xuất hiện bảng actor mới (VD: vendor).  Actor mới được quản lý bởi Auth Module.  Hiểu cách ánh xạ giữa auth\_identity và actor\_id.  . |  |
| **07/11/2025** | Test API login bằng Actor mới qua Postman/cURL.  Viết báo cáo tổng kết tuần. | Actor mới đã đăng nhập thành công.  Báo cáo tuần hoàn chỉnh. |  |
| **Tuần 7**  **10/11/2025** | Nghiên cứu kiến trúc Notification Module.  Đọc hiểu Notification Provider Module.  Phân tích interface AbstractNotificationProviderService (send, validate,…). | Nắm được luồng dữ liệu: Workflow → Notification Module → Provider.  Hiểu các yêu cầu khi tự viết Provider mới. |  |
| **11/11/2025** | Khởi tạo Custom Notification Provider.  Tạo cấu trúc thư mục module.  Định nghĩa Service kế thừa từ AbstractNotificationProviderService.  Viết hàm send() để console.log dữ liệu gửi đi.  . | Hoàn thành file service.ts của Provider.  Provider đã implement đầy đủ các phương thức bắt buộc. |  |
| **12/11/2025** | Đăng ký Provider mới vào medusa-config.ts.  Chạy server build/dev để kiểm tra module được load. | Server hoạt động ổn định, log hiển thị Provider mới đã được load.  Hiểu cách ánh xạ cấu hình từ medusa-config vào provider. |  |
| **13/11/2025** | Tạo workflow gửi thông báo thử nghiệm.  Sử dụng sendNotificationStep hoặc notificationModuleService.create().  Truyền template, to, data để test. | Workflow chạy đúng, log hiển thị dữ liệu thông báo.  Tính năng gửi thông báo cơ bản hoàn thiện. |  |
| **14/11/2025** | Debug lỗi tham số hoặc template ID.  Thử thay console.log bằng dịch vụ thật như Resend/SendGrid.  Viết báo cáo tổng kết tuần. | Workflow chạy và log ra console đúng thông tin. |  |
| **Tuần 8**  **17/11/2025** | Tổng hợp lại Auth Module + Notification Module.  Rà soát lại cấu hình container và module dependencies. | Hệ thống vận hành ổn định, không xung đột module. |  |
| **18/11/2025** | Thử Tích hợp Notification vào luồng xử lý Blogs: Ví dụ: Khi tạo Blog mới → gửi thông báo. | Trigger hoạt động đúng; notification được gửi đúng dữ liệu |  |
| **19/11/2025** | Kiểm thử lại Actor Module + Auth + Notification.  Test nhiều trường hợp lỗi khác nhau. | Các module phối hợp ổn định, không phát sinh lỗi hệ thống. |  |
| **20/11/2025** | Tiến hành nghiên cứu chuyên thêm về cơ chế phối hợp giữa ba thành phần: Auth Module – Actor – Notification Module trong MedusaJS.  Kiểm thử lại toàn bộ quy trình xác thực (authenticate → validate → issue token → map actor) để kiểm tra tính ổn định và sự tuân thủ theo thiết kế ban đầu. | Thu thập được bộ dữ liệu kiểm thử đầy đủ về hoạt động của Auth – Actor – Notification.  Hoàn thiện tài liệu kỹ thuật với các biểu đồ và mô tả chính xác dựa trên quan sát thực nghiệm, thay vì chỉ dựa vào lý thuyết. |  |
| **21/11/2025** | Tiến hành kiểm thử tổng hợp toàn bộ các tác vụ đã thực hiện trong các tuần 5–8, bao gồm:   * kiểm thử lại module Auth, * kiểm thử actor mới, * kiểm thử khả năng gửi thông báo qua Notification Module, * kiểm thử workflow tích hợp. | Rút ra được các điểm mạnh, điểm cần cải thiện và đề xuất được định hướng phát triển hệ thống cho những giai đoạn tiếp theo.. |  |
| **Tuần9**  **24/11/2025** | * Mở rộng module Blogs:   + Viết thêm "Moderation Step" để kiểm duyệt nội dung blog.   + Xây dựng bảng từ khóa cấm (bad words).   + Viết thuật toán rà soát nội dung:     - Tách từ khóa.     - Kiểm tra độ dài nội dung.     - Loại bỏ ký tự đặc biệt trước khi phân tích. * Gắn step kiểm duyệt vào workflow:   + Step 1: Validate input.   + Step 2: Content moderation.   + Step 3: Lưu blog.   + Step 4: Gửi notification.   **Khó khăn**   * Một số cấu trúc bị lỗi khi dùng Unicode ký tự tiếng Việt. * Cần tách logic ra service riêng để tránh file quá dài.   **Kết quả**   * Workflow Blogs chạy ổn định hơn. * Bổ sung tính năng thực tế vào hệ thống, tăng giá trị cho đồ án. * Có thể mở rộng thêm Dashboard thống kê từ khóa vi phạm. | Nắm được cách workflow hoạt động theo từng bước:   1. Khởi tạo context. 2. Thực thi từng step theo thứ tự. 3. Gọi các module tương ứng để xử lý dữ liệu. 4. Trả kết quả cuối cùng cho client.   Hiểu rõ cách gọi Notification bên trong workflow Blogs, phục vụ cho bước debug tuần sau.  Ghi chú lại luồng hoạt động chi tiết để đưa vào tài liệu kỹ thuật. |  |
| **25/11/2025** | **Phát triển Workflow Blogs nâng cao (tích hợp Content Moderation Step)**  **Công việc**   * Mở rộng module Blogs:   + Viết thêm "Moderation Step" để kiểm duyệt nội dung blog.   + Xây dựng bảng từ khóa cấm (bad words).   + Viết thuật toán rà soát nội dung:     - Tách từ khóa.     - Kiểm tra độ dài nội dung.     - Loại bỏ ký tự đặc biệt trước khi phân tích. * Gắn step kiểm duyệt vào workflow:   + Step 1: Validate input.   + Step 2: Content moderation.   + Step 3: Lưu blog.   + Step 4: Gửi notification.   **Khó khăn**   * Một số cấu trúc bị lỗi khi dùng Unicode ký tự tiếng Việt. * Cần tách logic ra service riêng để tránh file quá dài. | Workflow Blogs chạy ổn định hơn.  Bổ sung tính năng thực tế vào hệ thống, tăng giá trị cho đồ án.  Có thể mở rộng thêm Dashboard thống kê từ khóa vi phạm. |  |
| **27/11/2025** | **Tối ưu cấu hình, tái cấu trúc code và *tích hợp hệ thống Kinde Authentication***  **Công việc chung**   * Rà soát và tối ưu lại file medusa-config.ts:   + Gom các module cùng tính năng.   + Tách environment variables ra file .env.kinde.   + Tối ưu import để tránh vòng lặp dependency.   **Công việc trọng tâm: TÍCH HỢP KINDE AUTH**  (*Nội dung bổ sung theo yêu cầu*)  **1. Lý do tích hợp Kinde**   * Kinde là nền tảng Identity-as-a-Service hỗ trợ:   + Đăng nhập OAuth2 / OpenID Connect.   + Xác thực bằng Google, GitHub, Microsoft.   + Quản lý token và session ổn định hơn so với Email/Password truyền thống. * Mục tiêu: Nâng cao tính hiện đại của hệ thống và mở rộng khả năng đăng nhập.   **2. Các bước thực hiện**  **(a) Đăng ký dự án trong Kinde Dashboard**   * Tạo project mới. * Lấy các thông tin:   + KINDE\_CLIENT\_ID   + KINDE\_CLIENT\_SECRET   + KINDE\_ISSUER\_URL   + KINDE\_REDIRECT\_URL   + KINDE\_LOGOUT\_REDIRECT\_URL   **(c) Cài đặt package “medusa-auth-kinde” (tự viết hoặc tích hợp open-source)**   * Tạo provider kế thừa AbstractAuthModuleProvider. * Implement các hàm:   + authenticate()   + validateCallback()   + callback()   + verifyToken()   **(d) Thử nghiệm đăng nhập OAuth**   * Kiểm thử bằng:   + Google Sign-in.   + GitHub OAuth. * Kết quả token trả về đúng chuẩn OIDC:   + id\_token   + access\_token   + refresh\_token   **(e) Mapping User với Actor**   * Khi OIDC login → tự động tạo actor mới:   + type: customer   + hoặc vendor nếu login từ dashboard quản trị. * Lưu thông tin vào bảng:   + auth\_identity   + actor   + actor\_identity\_link   **3. Vấn đề gặp phải**   * Callback URL của Kinde phải khớp 100%, sai một ký tự sẽ báo lỗi “invalid redirect\_uri”. * Token của Kinde dùng thuật toán RS256, cần lấy public key để verify. | Đăng nhập bằng Kinde hoàn toàn thành công.  Token OIDC verify đúng.  Actor được tạo tự động khi user lần đầu đăng nhập.  Hệ thống có thể chuyển đổi giữa EmailPass provider và Kinde provider linh hoạt.  Đây là điểm nhấn kỹ thuật nổi bật trong báo cáo khóa luận. |  |
| **28/11/2025** | - Hoàn thiện phần tài liệu mô tả tích hợp Kinde vào hệ thống MedusaJS.  - Bổ sung mô tả kiến trúc tích hợp Kinde: vai trò của client, Kinde Auth, backend Medusa và Actor Module.  - Viết chi tiết luồng xử lý đăng nhập: quá trình người dùng đăng nhập, nhận mã xác thực, trao đổi token, xác thực và khởi tạo phiên đăng nhập.  - Mô tả cách hệ thống ánh xạ dữ liệu người dùng từ Kinde sang Actor nội bộ của Medusa (email, provider\_id, thông tin cá nhân...).  - Cập nhật lại tài liệu chính của đồ án: hoàn thiện chương liên quan đến Auth Module, Actor Module và Kinde Integration. | - Tài liệu tích hợp Kinde đã hoàn chỉnh và có thể đưa vào báo cáo thực tập.  - Nội dung mô tả rõ ràng, đầy đủ, không phụ thuộc sơ đồ, dễ đọc – dễ hiểu.  - Hệ thống minh bạch hơn về cơ chế đăng nhập, token flow và quy trình ánh xạ dữ liệu.  - Tài liệu chính đã được cập nhật đầy đủ, đảm bảo tính liền mạch và chuyên môn. |  |
| **Tuần 10**  **01/12/2025** | - Bắt đầu tuần làm việc với mục tiêu chính là xây dựng các module nâng cao cho hệ thống thương mại điện tử dựa trên MedusaJS.   - Tập trung tìm hiểu chuyên sâu về **cơ chế xác thực giỏ hàng (Cart Authentication)** trong Medusa, đặc biệt trong bối cảnh hệ thống đã được tích hợp Kinde tuần trước.   - Phân tích toàn bộ luồng hoạt động của giỏ hàng: từ khi người dùng thêm sản phẩm → tạo cart → cập nhật line-items → đồng bộ giá → yêu cầu thanh toán.   - Xác định các điểm then chốt yêu cầu xác thực đăng nhập nhằm tránh trường hợp người dùng thao tác trái phép, ví dụ:    • Gửi request thêm sản phẩm khi không đăng nhập.    • Sử dụng token giả mạo hoặc token đã hết hạn.    • Truy cập giỏ hàng của người dùng khác.   - Viết bản tài liệu thuyết minh nội bộ mô tả lại toàn bộ các bước xử lý để chuẩn bị cho việc triển khai code. | - Nắm rõ cấu trúc hoạt động của module cart trong Medusa.  - Xác định chính xác các API cần áp dụng cơ chế bảo vệ bằng token Kinde.  - Có tài liệu phân tích chi tiết phục vụ cho lập trình trong các ngày tiếp theo. |  |
| **02/12/2025** | - Chính thức bắt đầu triển khai **Cart Authentication Middleware** – một lớp bảo vệ kiểm tra token trong mọi request liên quan đến giỏ hàng.   - Tạo middleware xử lý các nhiệm vụ:    • Lấy token từ header Authorization.    • Gửi token đến Kinde để kiểm tra hợp lệ.    • Xác định user tương ứng trong Actor Module bên Medusa.    • Nếu không hợp lệ → chặn request ngay lập tức.   - Thiết lập danh sách các route bắt buộc kiểm tra token, chẳng hạn:    • POST /cart    • POST /cart/line-items    • GET /cart/:id   - Viết các cấu trúc thông báo lỗi chuẩn, tránh việc trả về thông điệp khó hiểu cho client. | - Hoàn thành phiên bản đầu tiên của middleware xác thực giỏ hàng.  - Toàn bộ route liên quan đến giỏ hàng đã có lớp kiểm tra bảo vệ.  - Chuẩn hóa phản hồi lỗi: thống nhất mã lỗi 401 và 403 theo đúng chuẩn API. |  |
| **03/12/2025** | - Dành trọn ngày để **kiểm thử module xác thực giỏ hàng** vừa hoàn thành hôm trước.   - Xây dựng hệ thống test gồm nhiều tình huống thực tế:    • Token hợp lệ → cho phép truy cập.    • Token hết hạn → chặn với lỗi 401.    • Token bị sửa hoặc giả mạo → chặn với lỗi 403.    • User tồn tại nhưng bị xóa khỏi hệ thống → chặn với lỗi 403.    • User cố gắng truy cập giỏ hàng của người khác.   - Sử dụng Postman kết hợp log server để xem chi tiết hoạt động của middleware.   - Sau khi chạy thử, phát hiện và tối ưu nhiều điểm:    • Giảm số lần gọi xác thực tới Kinde bằng caching token.    • Tối ưu câu trả lời API nhằm thống nhất format JSON. | - Luồng xác thực hoạt động ổn định.  - Xác định được vài điểm cần tối ưu: giảm số lần gọi Actor Module, cache metadata Kinde user.  - Middleware hoạt động chính xác trên hầu hết tình huống. |  |
| **04/12/2025** | - Bắt đầu nghiên cứu sâu vào **module khuyến mãi (Promotion Module)** – một trong những phần quan trọng nhất của hệ thống thương mại điện tử.   - Tìm hiểu các loại khuyến mãi mà hệ thống cần xử lý:    • Mã giảm giá theo phần trăm (%)    • Mã giảm giá theo số tiền cố định    • Khuyến mãi tự động theo sản phẩm hoặc nhóm sản phẩm    • Khuyến mãi theo tổng giá trị giỏ hàng   - Phân tích các hook mà Medusa hỗ trợ trong quá trình tính giá để chuẩn bị tích hợp khuyến mãi:    • beforeAddItem    • afterAddItem    • calculatePrice   - Viết tài liệu mô tả logic khuyến mãi, bao gồm điều kiện, cách tính và xử lý xung đột khi có nhiều khuyến mãi cùng lúc. | - Hoàn thành tài liệu phân tích yêu cầu chi tiết của module khuyến mãi.  - Xác định rõ toàn bộ các loại khuyến mãi cần hỗ trợ.  - Chuẩn bị tài liệu cho việc lập trình trong ngày 05/12. |  |
| **05/12/2025** | - Chính thức bước vào giai đoạn **lập trình module khuyến mãi**. Đây là ngày làm việc quan trọng nhất của tuần.   - Tạo các bảng (entity) cần thiết cho phía backend, bao gồm:    • promotions (mã giảm giá)    • promotion\_rules (quy tắc áp dụng)    • promotion\_usage (theo dõi số lượt dùng)   - Viết service chính xử lý khuyến mãi bao gồm:    • Kiểm tra xem mã giảm giá còn hiệu lực.    • Kiểm tra điều kiện đi kèm (đơn tối thiểu, sản phẩm áp dụng…).    • Tính toán số tiền giảm và cập nhật lại tổng giá trị giỏ hàng.   - Tạo API: POST /cart/apply-promo cho phép người dùng nhập mã giảm giá.   - Kiểm thử nhanh trên môi trường local để đảm bảo các tính năng hoạt động đúng mong đợi. | - Module khuyến mãi đã hoàn thành bản đầu tiên.  - Hệ thống đã hỗ trợ nhập mã giảm giá và tính toán giảm giá theo nhiều loại khác nhau.  - API apply promotion hoạt động ổn định.  - Chuẩn bị xong nền tảng để tiếp tục kiểm thử chuyên sâu vào các ngày sau. |  |

# CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ NGÂN HÀNG TMCP XUẤT NHẬP KHẨU

## 1.1. **Thông tin chung về đơn vị thực tập**

TVT GROUP được đầu tư và phát triển dựa trên nguyên tắc lựa chọn và tập trung, ngoài việc tiếp tục nâng cao thế mạnh vốn có của mình trong lĩnh vực Navigation, chúng tôi còn mở rộng thêm các hoạt động trên các lĩnh vực lập trình phần mềm ứng dụng cho các dòng sản phẩm, thiết bị di động thông minh (smart phone, tablet) sử dụng hệ điều hành Android, iOS, Window Phone, Blackberry OS… đào sâu nghiên cứu, cập nhật các công nghệ mới để cho ra đời các ứng dụng với nhiều tính năng thông minh, tiên tiến trên thế giới như công nghệ xử lý hình ảnh (Image Processing), Công nghệ tương tác thực tế ảo (Augmented Reality), DM (Direct marketing), DM+CRM…

|  |  |
| --- | --- |
| Tên công ty | **CÔNG TY CỔ PHẦN TVT GROUP VIETNAM** |
| Tel: | (08) 3715 0078 |
| Email: | contact.tvtgroup@gmail.com |
| Website: | http://www.tvtgroup.io/ |
| Gíam đốc đại diện: | Nguyễn Ngọc Tú |
| Địa chỉ: | Phòng 17, Nhà 8, CVPM Quang Trung, P. Tân Chánh Hiệp, Q12, TPHCM |

Bảng 1.. Chi tiết thông tin công ty



Hình 1.1. Logo CÔNG TY CỔ PHẦN TVT GROUP VIETNAM

## 1.2. Chức năng và Tầm nhìn

### 1.2.1. Chức năng của công ty

TVT GROUP được đầu tư và phát triển dựa trên nguyên tắc lựa chọn và tập trung, ngoài việc tiếp tục nâng cao thế mạnh vốn có của mình trong lĩnh vực Navigation, chúng tôi còn mở rộng thêm các hoạt động trên các lĩnh vực lập trình phần mềm ứng dụng cho các dòng sản phẩm, thiết bị di động thông minh (smart phone, tablet) sử dụng hệ điều hành Android, iOS, Window Phone, Blackberry OS… đào sâu nghiên cứu, cập nhật các công nghệ mới để cho ra đời các ứng dụng với nhiều tính năng thông minh, tiên tiến trên thế giới như công nghệ xử lý hình ảnh (Image Processing), Công nghệ tương tác thực tế ảo (Augmented Reality), DM (Direct marketing), DM+CRM…

### 1.2.2. Tầm nhìn của công ty

**Tầm nhìn**

Trở thành một đơn vị hàng đầu trong việc nghiên cứu và phát triển những phần mềm ứng dụng mang tính kỹ thuật cao, phù hợp với xu thế chung của nền công nghệ thế giới, khẳng định chất lượng sản phẩm và thương hiệu công ty tại thị trường Việt Nam nói riêng và thị trường châu Á nói chung.

Triết lý công ty: 5C

* **Challenge – Thách thức**

Thách thức cho các thành viên về việc luôn cập nhật công nghệ mới để thực hiện các dự án có yêu cầu cao. Thách thức cũng là cơ hội để các thành viên tự phát triển bản thân mình.

* **Collaboration – Hợp tác**

Hợp tác với các công ty đối tác hoặc khách hàng

* **Định hướng Khách hàng**

Hiểu rõ yêu cầu của khách hàng

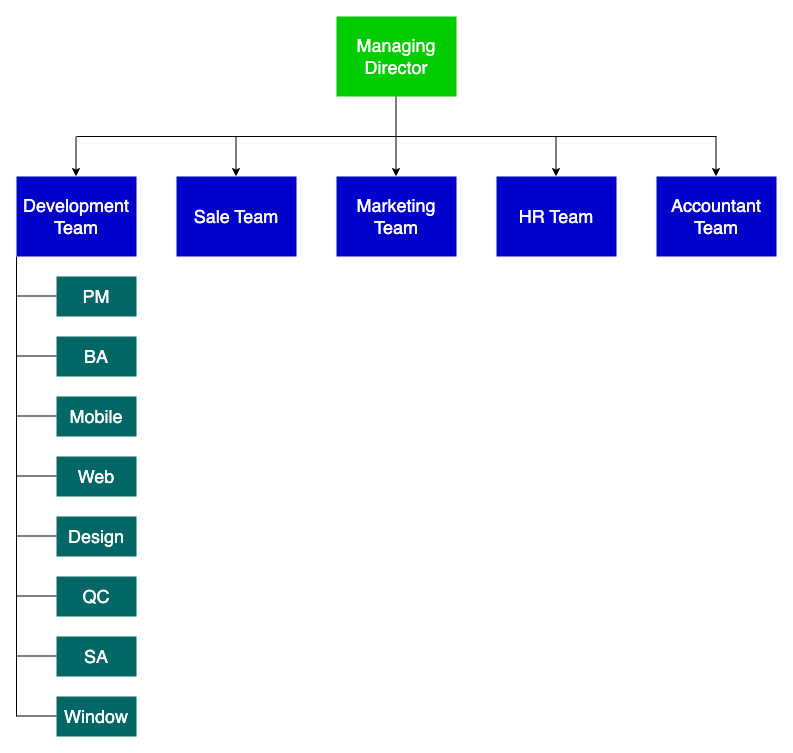
* **Tự tin**

Đảm bảo hiểu biết về công nghệ cũng như khả năng làm việc trong lĩnh vực đang đảm nhận

* **Thay đổi**

Ứng dụng kiến thức, công nghệ mới để liên tục thay đổi, đổi mới trong việc phát triển ý tưởng và ứng dụng.

**Cấu trúc tổ chức:**



**Làm việc theo đội**

Một tập thể làm việc hợp tác và hiệu quả là yếu tố thiết yếu cho sự thành công. Vì vậy Công ty luôn tạo điều kiện để các cá nhân làm việc theo đội, kết quả đạt được của đội chính là thành quả của mỗi cá nhân. Sự đóng góp và hợp tác của từng thành viên sẽ tạo ra sức mạnh của cả Công ty.

**Trao đổi thông tin**

Bạn được khuyến khích chia sẻ các quan điểm, tìm kiếm thông tin, cung cấp dữ liệu và giải quyết những vướng mắc cùng với cấp trên trực tiếp; khi cần thiết cũng có thể tham khảo ý kiến của bất kỳ thành viên nào trong Ban Lãnh đạo để đi đến giải pháp sau cùng. Trưởng nhóm của bạn luôn lắng nghe ý kiến của bạn, khuyến khích những nỗ lực và tìm giải pháp cho những khó khăn mà bạn gặp phải. Ban Giám đốc không chấp nhận cách truyền đạt thông tin sai lệch, đồn thổi, gây chia rẽ nội bộ làm tinh thần làm việc và quan hệ đồng nghiệp trở nên đố kỵ và thiếu hợp tác.

**Tôn trọng giá trị Nhân viên**

Công ty luôn xem con người là nguồn lực quý giá nhất. Vì thế, Công ty luôn tạo mọi điều kiện để bạn phát huy hết tiềm năng của mình bằng cách khuyến khích bạn học hỏi thêm và tin tưởng giao phó những trọng trách cao hơn, phù hợp với khả năng và kinh nghiệm mà bạn có thể đảm trách.

**Trách nhiệm**

Sự thành công của Công ty phụ thuộc vào việc thỏa mãn những quyền lợi hợp lý, đúng đắn của khách hàng và nhân viên. Điều này cần phải có sự hợp tác toàn tâm toàn ý của mọi thành viên trong Công ty. Bạn cần phải làm việc đúng giờ, đều đặn, hoàn thành công việc một cách hữu hiệu, đạt chất lượng cao và chấp hành nội qui, quy định và chuẩn mực đã đề ra.

**Tính chuyên nghiệp trong công việc**

Tính chuyên nghiệp được thể hiện bằng cam kết cung cấp giải pháp và công cụ cho khách hàng với chất lượng cao, dịch vụ hậu mãi chu đáo và uy tín, giao tiếp với khách hàng bằng thái độ hòa nhã, lịch thiệp, tôn trọng nhau trong thương lượng và tranh luận để đi đến thống nhất. Sử dụng các biểu mẫu, tài liệu chuẩn của công ty khi gửi đến khách hàng...

Mặt khác, tính chuyên nghiệp được thể hiện bằng việc chấp hành nghiêm túc nguyên tắc quản lý điều hành, đảm bảo hoàn thành xuất sắc nhiệm vụ được giao. Tính chuyên nghiệp cũng đòi hỏi bạn kiểm soát hữu hiệu cảm xúc cá nhân của mình để lắng nghe, trao đổi nhằm giải quyết các rắc rối, khó khăn trong công việc theo mục tiêu chung của công ty.

**Tự nguyện tham gia hoạt động xã hội và phát triển cộng đồng**

Ban Giám đốc và toàn thể nhân viên Công ty tự nguyện tham gia các hoạt động từ thiện hoặc các công tác xã hội để phát triển cộng đồng, tôn trọng pháp luật.

## 1.3. Quá trình hình thành và chính sách nhân sự của công ty

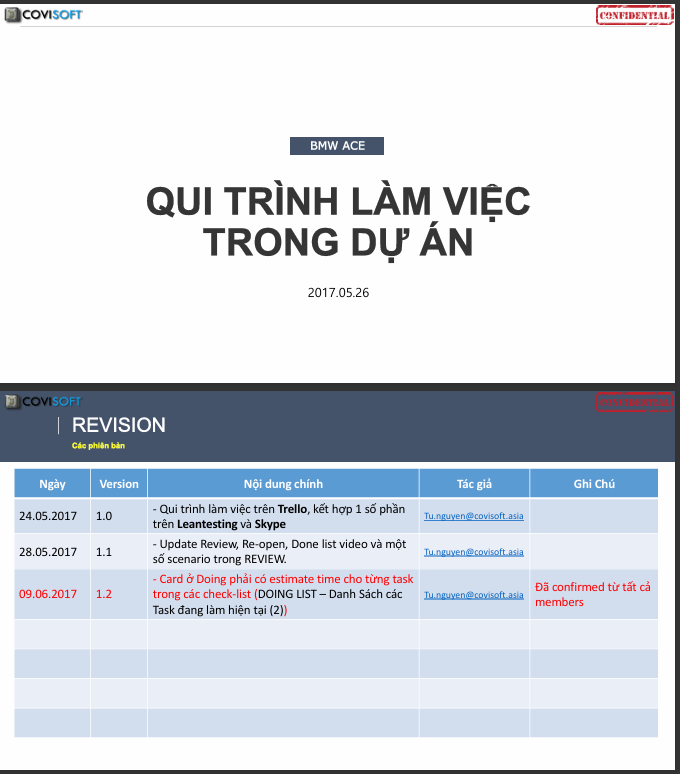
**Lịch sử TVT GROUP**

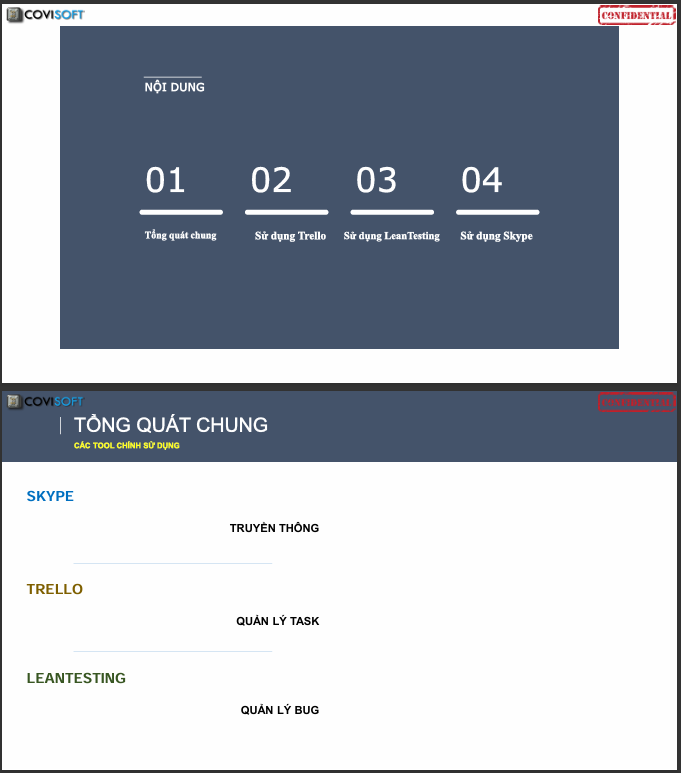
TVT GROUP tiền thân là công ty Covisoft– một đơn vị có nhiều năm kinh nghiệm trong lĩnh vực phát triển và nghiên cứu các ứng dụng về hệ thống định vị, chỉ dẫn (Navigation) tại Hàn Quốc

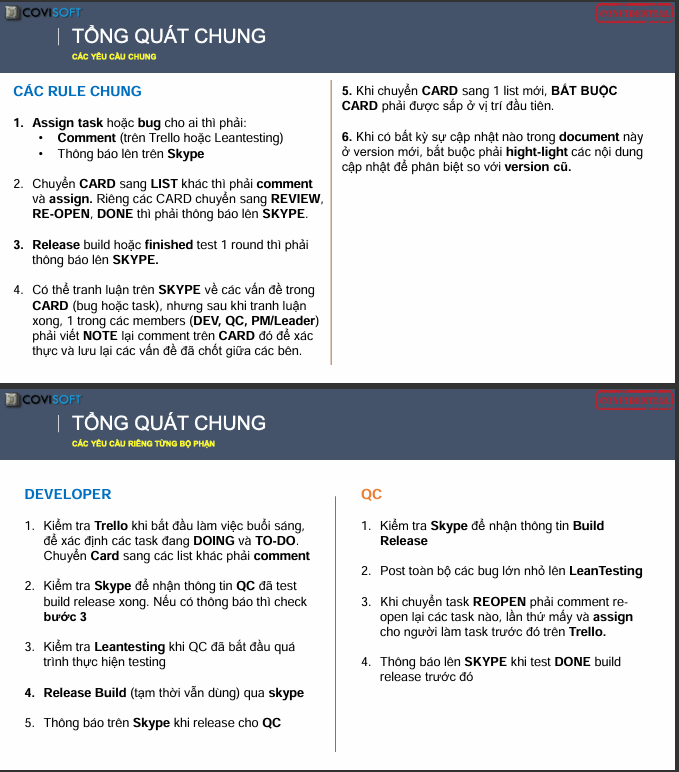
* **Covisoft Vietnam** được thành lập vào tháng 4/2012, với tên gọi Công ty TNHH Phần mềm Cồ Việt với 5 thành viên ban đầu. Hoạt động và phương hướng phát triển chính là O.D.C (Oversea Development Center)
  + Vào tháng 6/2014, Covisoft Vietnam đổi tên thành **Công ty TNHH Phần mềm Covisoft** do Ông Nguyễn Ngọc Tú đứng tên đại diện theo pháp luật.
* Tháng 08/2019, công ty đổi tên thành TVT GROUP để phục vụ các kế hoạch phát triển mở rộng trong tương lai.

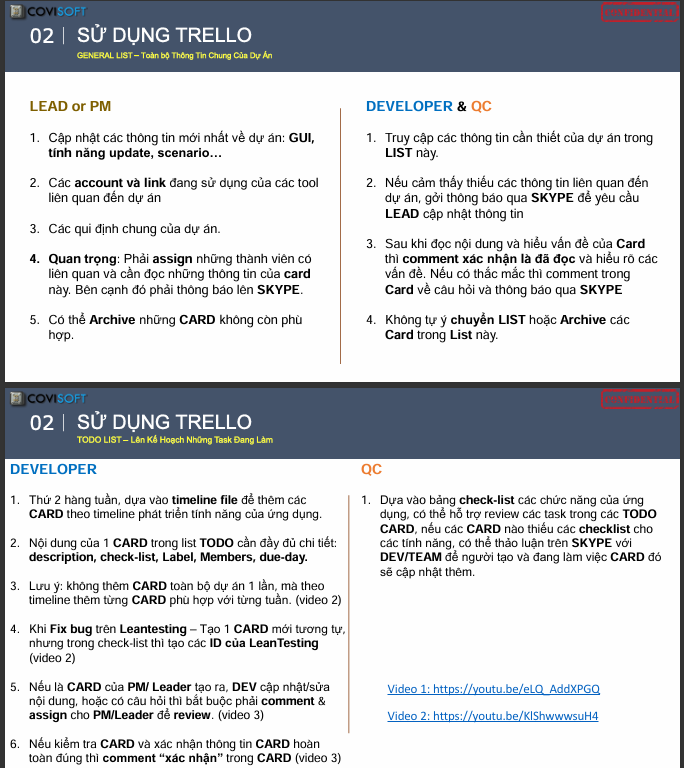
## 1.4. Quy trình làm việc & Quy định làm việc

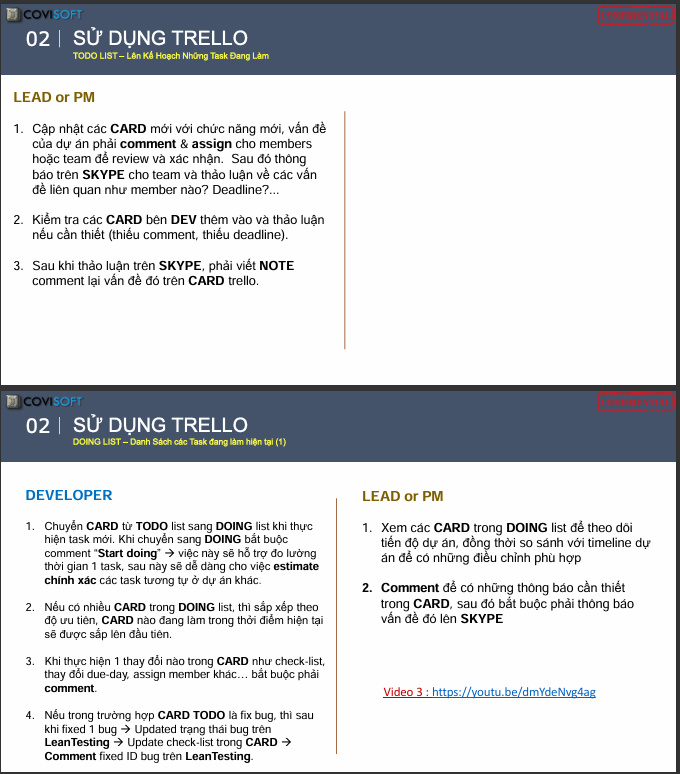
### 1.4.1. Quy trình làm việc

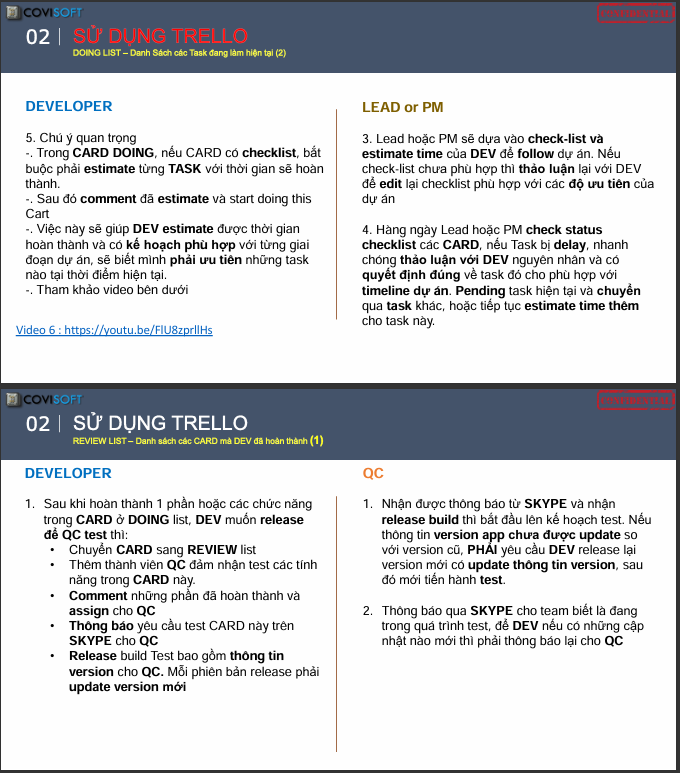


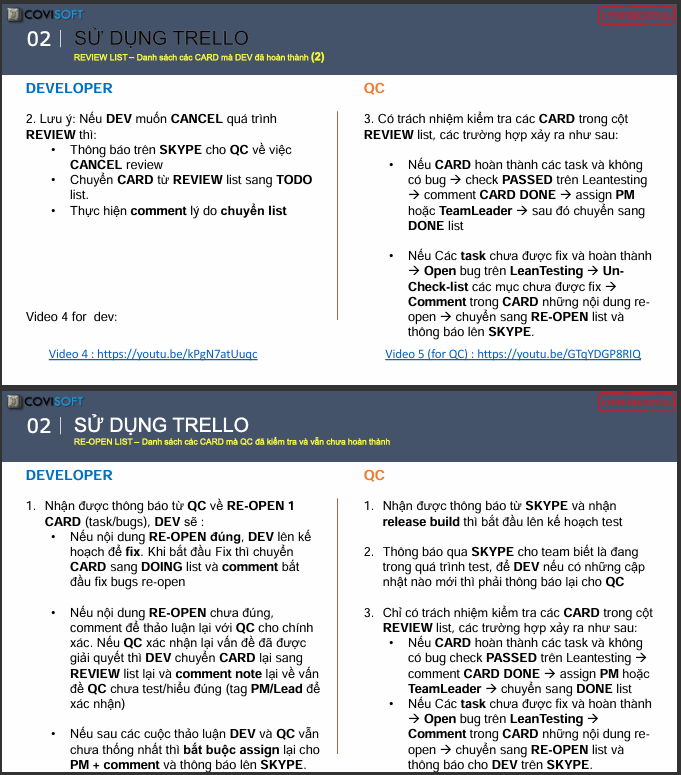


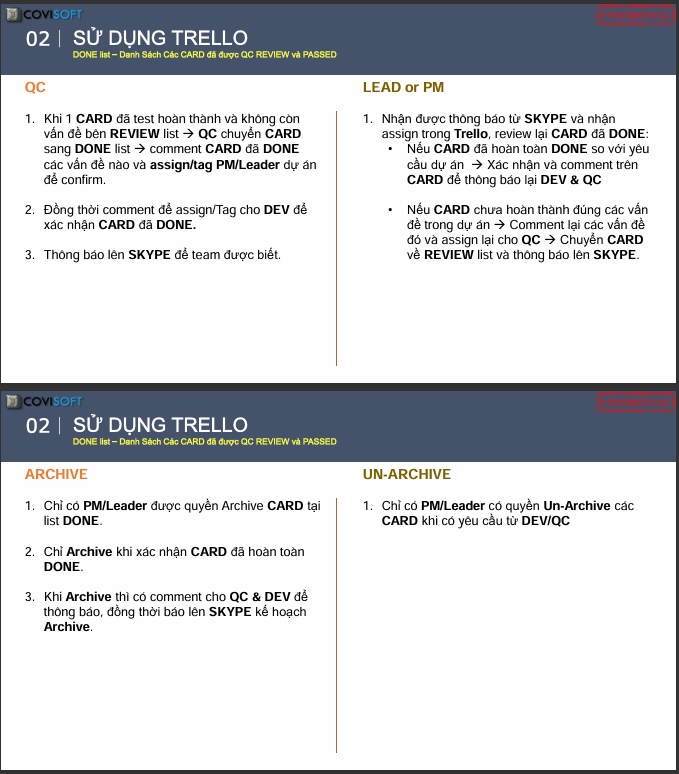
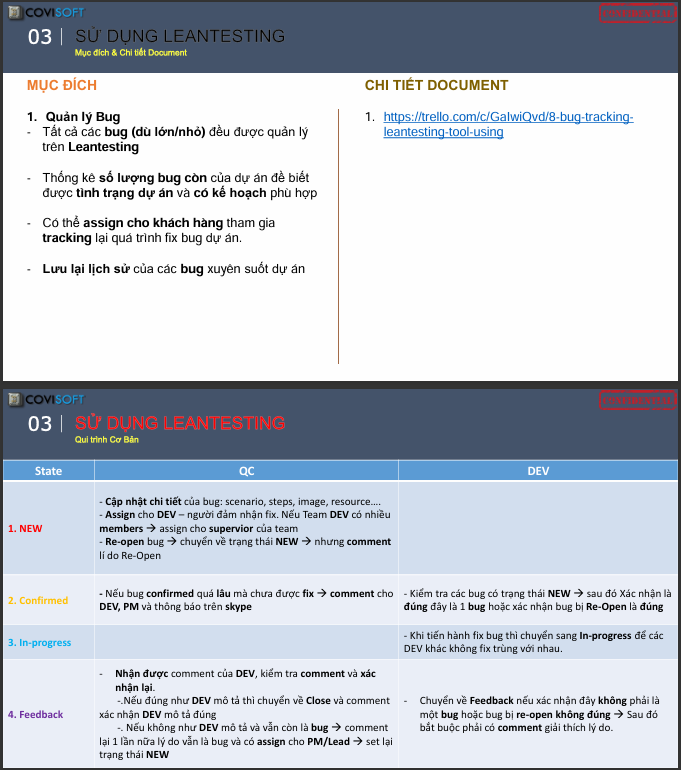


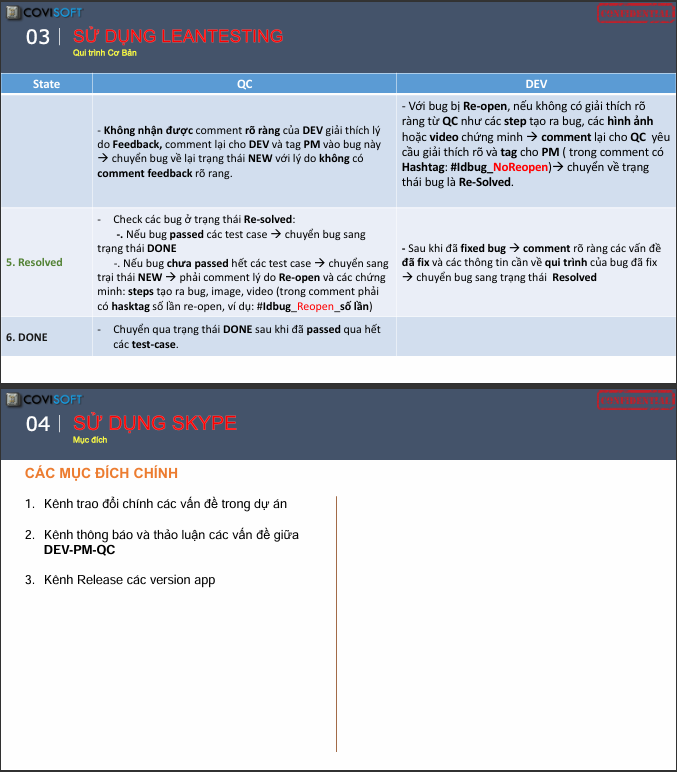










Hình 1.2.Quy trình làm việc trong dự án

### 1.4.2. Quy định làm việc

**TÓM TÁT MỘT SỐ QUY ĐỊNH ĐƯỢC THỐNG NHẤT**

**Revision History:**

1. **Quy trình làm việc trong dự án**

* Từ 2023 TVT GROUP sẽ apply toàn bộ các dự án theo qui trình Agile
* Team phát triển bao gồm BA, Design, dev, QC và PM họp thống nhất các yêu cầu cần hoàn thành trong phase / Sprints ( 2 tuần / 1 Sprint).
* Team phát triển lên kế hoạch, thời hạn hoàn thành các task trong phase.
* Team QC dựa vào kế hoạch của team dev, lên kế hoạch testplan, test case ...
* PM cùng với team dev và team QC thống nhất thêm khoản thời gian dự trữ để fix bug.
* Nếu team dev cảm thấy tiến độ chậm thì tự giác đề xuất với PM để tiến hành OT, cố gắng hoàn thành trước deadline.
* Nếu team dev đã nỗ lực nhưng biết không thể hoàn thành trước deadline, phải thông báo và giải thích kịp thời cho PM.
* Các team sẽ họp để kết thúc 1 sprint và lên kế hoạch lại vào mỗi đầu Sprint.
* Tool áp dụng cho việc quản lý dự án sẽ là JIRA, các hoạt động còn lại của Team sẽ diễn ra trên LARK

1. **Task khẩn cấp**

* Khi có task khẩn cấp phát sinh, PM sẽ yêu cầu team dev imlement hoặc team QC test.
* Các team sẽ ưu tiên giải quyết task khẩn cấp, lập lại kế hoạch và chú thích lý do.

1. **Report**

* Khi release một phiên bản phần mềm, phải kèm theo tài liệu report những tính năng mới và những bug đã fix.
* Khi QC test xong, viết bản report những tính năng đã test, và báo lỗi trên JIRA.

1. **Làm việc với QC**

* Mỗi project chỉ có một QC đảm nhiệm chính. Kế hoạch test do nội bộ team QC tự sắp xếp. PM và dev chỉ cần trao đổi trực tiếp với QC đảm nhiệm chính.
* Cuộc họp nào liên quan đến scenario, design, tính năng thì phải có QC tham gia.

1. **Tạo và quản lý Dự án ở JIRA**

**LÀM VIỆC DỰA TRÊN DESIGN**

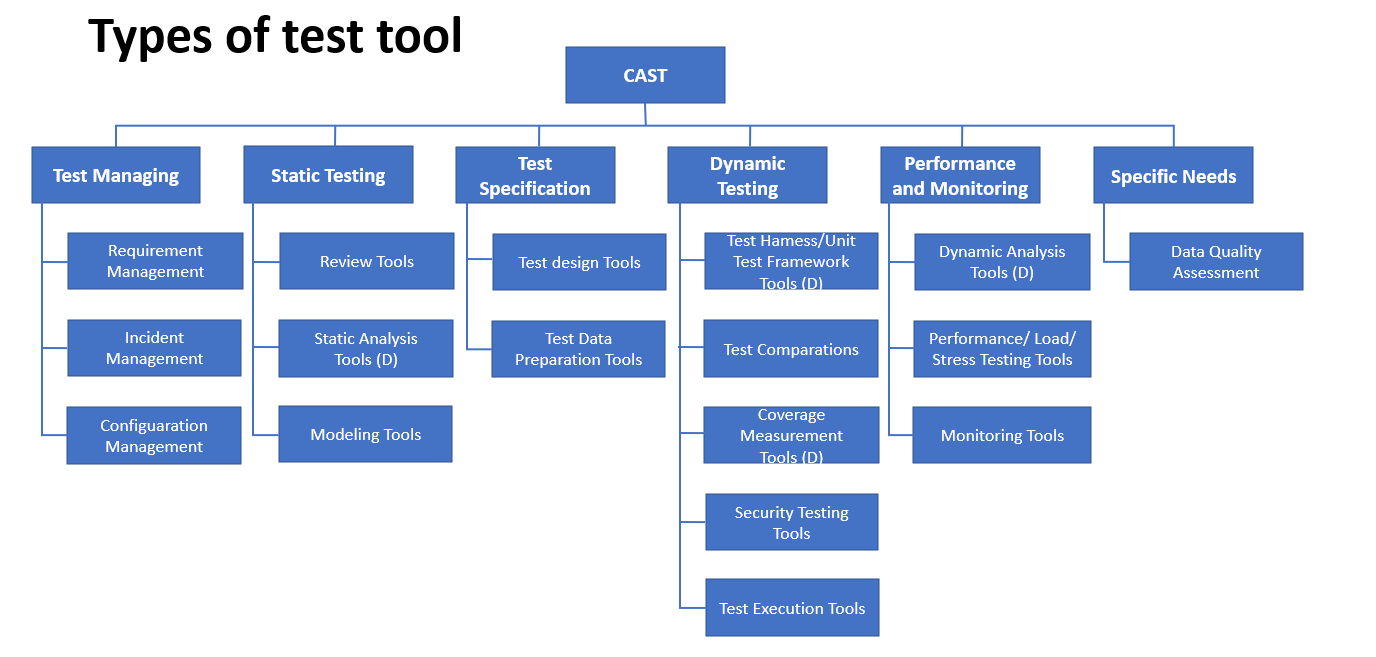
**Revision History:**

1. **Quy trình làm việc trong dự án theo Desing**

* DEV iOS và Android follow theo UI design đã được thống nhất và approved
* QC là người kiểm định lại dev bên nào không follow theo design và bắt buộc sửa
* Trong quá trình implement, iOS và Android dev nếu cảm thấy design hiện tại kĩ thuật không đáp ứng được trong nền tang của mình --> hãy yêu cầu với PM và thống nhất với dev còn lại sao cho iOS và Android phải chung 1 design (nếu đc thay đổi). QC sẽ follow theo để test
* Bất cứ sự thay đổi nào xuất phát từ ý tưởng cá nhân, không khớp với design thì bên bộ phận đó chịu trách nhiệm sửa đổi như design

## 1.5. Quy trình kiểm thử của team QC

|  |  |
| --- | --- |
| **Domain** | **Details** |
| **Fundamentals of Testing** | |
| Why is Testing Necessary? | - Describe, with examples, the way in which a defect in software can cause harm to a person, to the environment or to a company  - Distinguish between the root cause of a defect and its effects  - Give reasons why testing is necessary by giving examples  - Describe why testing is part of quality assurance and give examples of how testing contributes to higher quality  - Explain and compare the terms error, defect, fault, failure and the corresponding terms mistake and bug, using examples |
| What is Testing? | - Recall the common objectives of testing  - Provide examples for the objectives of testing in different phases of the software life cycle  - Differentiate testing from debugging |
| Seven Testing Principles | - Explain the seven principles of testing |
| Fundamental Test Process | - Recall the five fundamental test activities and respective tasks from planning to closure |
| The Psychology of Testing | - Recall the psychological factors that influence the success of testing  - Contrast the mindset of a tester and of a developer |
| **Testing Throughout the Software Life Cycle** | |
| Software Development Models | - Explain the relationship between development, test activities and work products in the development life cycle, by giving examples using project and product types  - Recognize the fact that software development models must be adapted to the context of project and product characteristics  - Recall characteristics of good testing that are applicable to any life cycle model |
| Test Levels | - Compare the different levels of testing: major objectives, typical objects of testing, typical targets of testing (e.g., functional or structural) and related work products, people who test, types of defects and failures to be identified |
| Test Types | - Compare four software test types (functional, non-functional, structural and change-related) by example  - Recognize that functional and structural tests occur at any test level  - Identify and describe non-functional test types based on non-functional requirements  - Identify and describe test types based on the analysis of a software system's structure or architecture  - Describe the purpose of confirmation testing and regression testing |
| Maintenance Testing | - Compare maintenance testing (testing an existing system) to testing a new application with respect to test types, triggers for testing and amount of testing  - Recognize indicators for maintenance testing (modification, migration and retirement)  - Describe the role of regression testing and impact analysis in maintenance |
| **Static Techniques** | |
| Static Techniques and the Test Process | - Recognize software work products that can be examined by the different static techniques  - Describe the importance and value of considering static techniques for the assessment of software work products  - Explain the difference between static and dynamic techniques, considering objectives, types of defects to be identified, and the role of these techniques within the software life cycle |
| Review Process | - Recall the activities, roles and responsibilities of a typical formal review  - Explain the differences between different types of reviews: informal review, technical review, walkthrough and inspection  - Explain the factors for successful performance of reviews |
| Static Analysis by Tools | - Recall typical defects and errors identified by static analysis and compare them to reviews and dynamic testing  - Describe, using examples, the typical benefits of static analysis  - List typical code and design defects that may be identified by static analysis tools |
| **Test Design Techniques** | |
| The Test Development Process | - Differentiate between a test design specification, test case specification and test procedure specification  - Compare the terms test condition, test case and test procedure  - Evaluate the quality of test cases in terms of clear traceability to the requirements and expected results  - Translate test cases into a well-structured test procedure specification at a level of detail relevant to the knowledge of the testers |
| Categories of Test Design Techniques | - Recall reasons that both specification-based (black-box) and structure-based (white-box) test design techniques are useful and list the common techniques for each  - Explain the characteristics, commonalities, and differences between specification-based testing, structure-based testing and experience-based testing |
| Specification-based or Black-box Techniques | - Write test cases from given software models using equivalence partitioning, boundary value analysis, decision tables and state transition diagrams/tables  - Explain the main purpose of each of the four testing techniques, what level and type of testing could use the technique, and how coverage may be measured  - Explain the concept of use case testing and its benefits |
| Structure-based or White-box Techniques | - Describe the concept and value of code coverage  - Explain the concepts of statement and decision coverage, and give reasons why these concepts can also be used at test levels other than component testing (e.g., on business procedures at system level)  - Write test cases from given control flows using statement and decision test design techniques  - Assess statement and decision coverage for completeness with respect to defined exit criteria. |
| Experience-based Techniques | - Recall reasons for writing test cases based on intuition, experience and knowledge about common defects  - Compare experience-based techniques with specification-based testing techniques |
| Choosing Test Techniques | - Classify test design techniques according to their fitness to a given context, for the test basis, respective models and software characteristics |
| **Test Management** | |
| Test Organization | - Recognize the importance of independent testing  - Explain the benefits and drawbacks of independent testing within an organization  - Recognize the different team members to be considered for the creation of a test team  - Recall the tasks of typical test leader and tester |
| Test Planning and Estimation | - Recognize the different levels and objectives of test planning  - Summarize the purpose and content of the test plan, test design specification and test procedure documents according to the 'Standard for Software Test Documentation'  - Differentiate between conceptually different test approaches, such as analytical, model-based, methodical, process/standard compliant, dynamic/heuristic, consultative and regression-averse  - Differentiate between the subject of test planning for a system and scheduling test execution  - Write a test execution schedule for a given set of test cases, considering prioritization, and technical and logical dependencies  - List test preparation and execution activities that should be considered during test planning  - Recall typical factors that influence the effort related to testing  - Differentiate between two conceptually different estimation approaches: the metrics-based approach and the expert-based approach  - Recognize/justify adequate entry and exit criteria for specific test levels and groups of test cases |
| Test Progress Monitoring and Control | - Recall common metrics used for monitoring test preparation and execution  - Explain and compare test metrics for test reporting and test control (e.g., defects found and fixed, and tests passed and failed) related to purpose and use  - Summarize the purpose and content of the test summary report document according to the 'Standard for Software Test Documentation' |
| Configuration Management | - Summarize how configuration management supports testing |
| Risk and Testing | - Describe a risk as a possible problem that would threaten the achievement of one or more stakeholders' project objectives  - Remember that the level of risk is determined by likelihood (of happening) and impact (harm resulting if it does happen)  - Distinguish between the project and product risks  - Recognize typical product and project risks  - Describe, using examples, how risk analysis and risk management may be used for test planning |
| Incident Management | - Recognize the content of an incident report according to the 'Standard for Software Test Documentation'  - Write an incident report covering the observation of a failure during testing |
| **Tool Support for Testing** | |
| Types of Test Tools | - Classify different types of test tools according to their purpose and to the activities of the fundamental test process and the software life cycle  - Explain the term test tool and the purpose of tool support for testing |
| Effective Use of Tools: Potential Benefits and Risks | - Summarize the potential benefits and risks of test automation and tool support for testing  - Remember special considerations for test execution tools, static analysis, and test management tools |
| Introducing a Tool into an Organization | - State the main principles of introducing a tool into an organization  - State the goals of a proof-of-concept for tool evaluation and a piloting phase for tool implementation  - Recognize that factors other than simply acquiring a tool are required for good tool support |



Hình 1.3. Types of test tool

7 Nguyên tắc kiểm thử phần mềm:  
**-Nguyên tắc 1 – Kiểm thử đưa ra lỗi**

Kiểm thử có thể cho thấy rằng phần mềm đang có lỗi, nhưng không thể chứng minh rằng phần mềm không có lỗi. Kiểm thử làm giảm xác suất lỗi chưa tìm thấy vẫn còn trong phần mềm, thậm chí là không còn lỗi nào, nó không phải là bằng chứng của sự chính xác.

**-Nguyên tắc 2 – Exhaustive testing is impossible**

Kiểm thử mọi thứ (tất cả các tổ hợp của điều kiện input đầu vào) là không thể thực hiện được, trừ phi nó chỉ bao gồm một số trường hợp bình thường (ít trường hợp tổ hợp thì có thể test toàn bộ được). Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên sự mức độ ưu tiên chúng ta có thể tập trung việc kiểm thử vào một số điểm cần thiết.

**-Nguyên tắc 3 – Kiểm thử sớm**

Để tìm được bug sớm, các hoạt động kiểm thử nên được bắt đầu càng sớm càng tốt trong qui trình phát triển (vòng đời phát triển) phần mềm hoặc hệ thống, và nên tập trung vào các hoạt động đã định trước.

**-Nguyên tắc 4 – Sự tập trung của lỗi**

Nỗ lực kiểm thử nên tập trung một cách cân đối vào mật độ lỗi dự kiến và lỗi phát hiện ra sau đó trong các mô-đun. Một số ít các mô-đun thường chứa nhiều lỗi không phát hiện ra trong lúc kiểm thử trước khi phát hành (release), hoặc chịu trách nhiệm cho hầu hết các lỗi hoạt động của phần mềm.

Để hiểu rõ hơn nguyên tắc này, ta cần xem xét 3 điều sau:

**1. Nguyên tắc tổ gián:** chỗ nào có 1 vài con gián thì ở đâu đó xung quanh nó sẽ có cả tổ gián -> có rất nhiều gián >>> chỗ nào có 1 vài con bug thì xung quanh, gần gần chỗ đó sẽ có nhiều bug.

**2. Nguyên tắc 80/20:** thông thường 20% chức năng quan trọng trong một chương trình có thể gây ra đến 80% tổng số bug phát hiện được trong chương trình đó.

**3. Exhaustive testing is impossible (nguyên tắc thứ 2):** do đó cần phải ananlysis (phân tích) + priorities (tính toán mức độ ưu tiên) để quyết định tập trung vào test chỗ nào.

=> Test kỹ chức năng quan trọng => found bug => test những gì liên quan + những chức năng gần nó để tìm ra bug nhiều hơn.

**-Nguyên tắc 5 – Nghịch lý thuốc trừ sâu**

Nếu việc kiểm thử tương tự nhau được lặp đi lặp lại nhiều lần, thì cuối cùng sẽ có một số trường hợp kiểm thử (ca kiểm thử - test case) sẽ không còn tìm thấy bất kỳ lỗi nào mới. Để khắc phục "nghịch lý thuốc trừ sâu" này, các trường hợp kiểm thử cần phải được xem xét và sửa đổi thường xuyên, và cần phải viết các test case mới và khác nhau để thực hiện nhiều phần khác nhau của phần mềm hoặc hệ thống để tìm ra lỗi tiềm ẩn nhiều hơn nữa.

Nguyên tắc này giống như việc trừ sâu trong nông nghiệp, nếu chúng ta cứ phun một loại thuốc với nồng độ giống nhau trong một khoảng thời gian dài thì có một số con sâu sẽ quen dần và cuối cùng việc phun thuốc giống như là tắm chúng vậy (bị lờn thuốc) => lúc đó chúng ta không thể diệt sạch chúng được. Do vậy, để diệt sạch sâu một cách hiệu quả, người ta thường thay đổi loại thuốc trừ sâu, mỗi loại chỉ dùng trong khoảng thời gian ngắn.

**-Nguyên tắc 6 – Kiểm thử theo các ngữ cảnh độc lập**

Nguyên tắc này là việc testing phụ thuộc vào ngữ cảnh, test trong nhiều ngữ cảnh khác nhau.

Để hiểu rõ hơn chúng ta xem ví dụ sau:

Ví dụ, cũng với một chương trình calculator có rất nhiều chức năng, nhưng:

* Nếu test chương trình này cho mẫu giáo thì chỉ cần test cộng trừ là OK
* Nếu test chương trình này cho cấp 2 thì cộng trừ nhân chia
* Nếu test chương trình này cho đại học thì tích phân, đạo hàm, v.v....

**-Nguyên tắc 7 – Sự sai lầm về việc không có lỗi**

Việc tìm và sửa chữa lỗi sẽ không giúp được gì nếu hệ thống được xây dựng xong nhưng không thể dùng được và không đáp ứng được nhu cầu và sự mong đợi của người dùng. (Nghĩa là nếu sau khi code, test rồi fix bug, làm đủ tất cả các trường hợp và cuối cùng cho ra một sản phẩm không như mong đợi hoặc không đáp ứng được nhu cầu của khách hàng thì dự án phần mềm đó coi như thất bại mặc dù đã được test xong)

## 1.6. **Quy trình test phần mềm trong quá trình sản xuất phần mềm.**

Sau khi nhận được tài liệu yêu cầu (spec) từ khách hàng hoặc bộ phận thiết kế chi tiết.

QC sẽ tìm hiểu spec và viết test plan, sau đó gửi test plan cho khách hàng và project manager xem lại, sau khi duyệt test plan này.

QC sẽ tiến hành viết test case dựa vào spec đó. (song song đó DEV sẽ dựa vào spec để code)

Nếu công ty có bộ phận test whitebox thì dựa vào spec viết test case test whitebox và viết test script để test whitebox. (thường gọi là unit test - ở đây, "unit" là 1 hàm, 1 class hoặc 1 component,... tùy cách nhìn nhận và quản lý của mỗi công ty)

Sau khi test whitebox thành công (test pass các test case whitebox) thì QC sẽ tiến hành test blackbox (test chức năng của từng màn hình - cái này cũng được xem là unit test, "unit" ở đây là 1 màn hình hoặc một chức năng tùy qui định và cách quản lý của mỗi công ty).

Sau khi test từng màn hình thành công (pass hết các test case hoặc đạt được số % test case pass nào đó - ví dụ 97% test case pass) thì chuyển sang giai đoạn test tích hợp (integration testing) là kết hợp một số màn hình/chức năng có liên quan lại với nhau rồi test theo luồng xử lý (user story)

Sau khi pass vòng này thì sẽ tiến hành tổng hợp toàn bộ hệ thống (sản phẩm phần mềm) và tiến hành test ở mức hệ thống.

Có một số công ty hoặc khách hàng yêu cầu test UAT (User Acceptance Testing) thì sẽ thực hiện test lại hệ thống theo các chức năng đã được mô tả trong spec.

Có một số loại phần mềm hoặc khách hàng yêu cầu hoặc qui trình sản xuất phần mềm của công ty, phần mềm sẽ được test alpha và beta.

Trên đây là một qui trình test 1 phần mềm. Tùy vào tính chất của sản phẩm và qui trình sản xuất của mỗi công ty, qui trình trên có thể có nhiều hoặc ít loại test khác hơn.

Trong quá trình test ở bất kỳ mức nào, nếu phát hiện bug thì tester sẽ post bug lên hệ thống quản lý bug (bằng excel hoặc chương trình riêng) và DEV sẽ dựa vào đó fix bug và tester sẽ test lại. (đây là qui trình quản lý và xử lý bug - sẽ nói riêng ở một bài khác)

# CHƯƠNG 2: NỘI DUNG THỰC TẬP TẠI CÔNG TY CỔ PHẦN TVT GROUP

## 2.1. Giới thiệu về luồng dữ liệu:

Trong quá trình thực tập, hệ thống được xây dựng dựa trên kiến trúc module của MedusaJS. Luồng dữ liệu trong hệ thống được tổ chức theo mô hình đa tầng, đảm bảo dễ mở rộng, dễ bảo trì và hỗ trợ tích hợp với nhiều dịch vụ ngoài như Kinde Authentication hay CMS.

Luồng dữ liệu tổng quát bao gồm:

* **Client → API Gateway → Module Layer → Service Layer → Database**
* Mỗi thao tác của người dùng (đăng nhập, tạo blog, cập nhật giỏ hàng…) sẽ đi qua nhiều lớp xử lý:
  1. **Route / API Handler** – tiếp nhận request
  2. **Middleware** – xác thực, kiểm tra token, hạn chế truy cập trái phép
  3. **Workflow / Module** – xử lý logic chính của tính năng
  4. **Service** – truy vấn và cập nhật dữ liệu
  5. **Database** – lưu trữ kết quả cuối

Đặc biệt, sau tuần 9–10, luồng dữ liệu trở nên phức tạp hơn với các thành phần:

* **Auth Module + Actor Module** (xác thực, ánh xạ người dùng)
* **Kinde Authentication** (OAuth2/OIDC token flow)
* **Cart Authentication Middleware** (kiểm soát mọi thao tác giỏ hàng)
* **Promotion Workflow** (tính toán mã giảm giá, xử lý xung đột khuyến mãi)

Nhờ đó, hệ thống được bảo vệ tốt hơn và đáp ứng yêu cầu thương mại điện tử thực tế.

## 2.2. Công cụ hỗ trợ phát triển – MedusaJS Framework

Trong suốt thời gian thực tập, MedusaJS đóng vai trò là framework chính để xây dựng hệ thống thương mại điện tử. MedusaJS cung cấp:

* **Hệ thống module hóa (Modular Architecture)** cho phép mở rộng bằng Auth, Actor, Notification, Blogs, Promotion…
* **Service Container & Dependency Injection** giúp quản lý các service rõ ràng và tách biệt.
* **Workflow Engine** cho phép xây dựng chuỗi xử lý theo từng bước (ví dụ: Blogs Workflow, Notification Workflow).
* **Hỗ trợ API REST tiêu chuẩn**, dễ mở rộng cho các chức năng mới.
* **Hỗ trợ tích hợp với các dịch vụ ngoài** như CMS, Kinde, Email Provider.

MedusaJS giúp hệ thống phát triển theo hướng linh hoạt, dễ cấu hình và phù hợp cho các dự án thương mại điện tử hiện đại.

## 2.3. Công cụ và nền tảng xác thực – Kinde Authentication

Nhằm hiện đại hóa hệ thống và thay thế phương thức Email/Password truyền thống, đề tài đã tích hợp **Kinde Authentication** – nền tảng Identity-as-a-Service.

Các khả năng chính của Kinde:

* **Hỗ trợ OAuth2 / OpenID Connect**
* Đăng nhập qua **Google, GitHub, Microsoft**
* Cung cấp **id\_token, access\_token, refresh\_token**
* Quản lý phiên đăng nhập ổn định
* Dễ dàng ánh xạ người dùng sang Actor trong Medusa

Kinde giúp hệ thống đạt được:

* Xác thực an toàn
* Quản lý token theo chuẩn OIDC
* Dễ mở rộng cho các ứng dụng lớn
* Tích hợp linh hoạt với Auth Module của MedusaJS

Đây là một điểm nhấn kỹ thuật quan trọng trong suốt quá trình thực tập.

## 2.4. Thực tế quy trình tiếp nhận và xử lý yêu cầu lấy dữ liệu tại Eximbank

Dựa trên quy trình làm việc thực tế trong 10 tuần, nhóm đã hình thành một chu trình xử lý yêu cầu gồm bốn giai đoạn chính.

### 2.4.1. Khởi tạo và gửi yêu cầu

Trong mỗi tuần, yêu cầu mới được đặt ra dựa trên:

* Mục tiêu tuần
* Kế hoạch nghiên cứu
* Tính năng cần hoàn thiện (Auth, Notification, Blogs, Kinde, Cart Auth, Promotion…)

Các bước chính gồm:

* Thu thập yêu cầu
* Phân tích luồng hoạt động
* Viết tài liệu mô tả
* Xác định các module cần tạo mới hoặc mở rộng

Ví dụ:

* Tuần 6: phân tích Auth Module và Actor
* Tuần 9: phân tích Blogs Workflow + Moderation
* Tuần 10: phân tích giỏ hàng + khuyến mãi

### 2.4.2. Tiếp nhận và thẩm định yêu cầu kỹ thuật

Sau khi khởi tạo yêu cầu, bước tiếp theo là thẩm định khả thi kỹ thuật:

* Xác định framework hỗ trợ không
* Kiểm tra module liên quan có xung đột hay không
* Đánh giá mức độ ảnh hưởng tới hệ thống
* Xác định cơ chế bảo mật hoặc xác thực cần bổ sung

Ví dụ:

* Khi tích hợp Kinde → cần xác minh token RS256, cấu hình callback URL
* Khi xây dựng Cart Middleware → phải đảm bảo token hợp lệ trước khi truy cập giỏ hàng
* Khi viết Blogs Moderation → phải tách logic ra service để tránh file quá dài

### 2.4.3. Triển khai, trích xuất và xử lý dữ liệu

Đây là giai đoạn lập trình và kiểm thử chính:

* Tạo module mới (Blogs, Notification Provider, Promotion…)
* Xây dựng service logic
* Kết nối database (tạo migrations)
* Viết route và middleware
* Chạy workflow theo từng step
* Kiểm thử bằng Postman
* Log server để theo dõi dữ liệu chạy qua từng lớp

Ví dụ nổi bật:

* OAuth Login với Kinde
* Cart Authentication với token OIDC
* Moderation Workflow kiểm tra bad words
* Promotion Module tính toán giảm giá

### 2.4.4. Bàn giao kết quả và đóng yêu cầu

Sau khi kiểm thử hoàn tất:

* Tổng hợp tài liệu kỹ thuật
* Viết báo cáo tuần
* Ghi chú lại luồng xử lý
* Chốt kết quả và chuẩn bị bước nâng cấp tiếp theo

Nhờ quy trình tuần–theo–tuần, hệ thống ngày càng hoàn thiện và đáp ứng đầy đủ yêu cầu của đồ án.

## 2.5. Thực tế quy trình tiếp nhận và xử lý yêu cầu lấy dữ liệu tại Eximbank

Qua 10 tuần thực tập, các kết quả đạt được gồm:

**1. Nắm vững kiến trúc MedusaJS**

* Hiểu container, module, workflows, service
* Tự xây dựng module Blogs, Notification Provider, Promotion

**2. Hoàn thiện hệ thống xác thực nâng cao**

* Tích hợp thành công Kinde Authentication (Google, GitHub login)
* Xây dựng Actor mới (vendor, shipper)
* Kết nối Actor & Auth Module

**3. Tạo và triển khai nhiều tính năng thực tế**

* Notification Module tùy chỉnh
* Blogs Workflow + Moderation
* Cart Authentication Middleware
* Promotion Module với đầy đủ logic

**4. Kỹ năng kiểm thử và debug**

* Kiểm thử API bằng Postman
* Xây dựng test scenario
* Tối ưu hiệu năng và caching token

**5. Kỹ năng viết tài liệu chuyên nghiệp**

* Tài liệu Auth Module
* Tài liệu Kinde Integration
* Tài liệu Cart Authentication
* Tài liệu Promotion Logic
* Slide báo cáo và tổng kết tuần

# KẾT LUẬN

Qua quá trình thực tập tại dự án phát triển hệ thống thương mại điện tử trên nền tảng MedusaJS, em đã đạt được nhiều kết quả đáng kể cả về kiến thức lý thuyết lẫn kỹ năng thực hành.

Cụ thể:

1. **Nắm vững kiến trúc hệ thống và luồng dữ liệu:**
   * Hiểu rõ cơ chế hoạt động của các module trong MedusaJS, bao gồm container, service, module loader và workflow.
   * Nắm vững luồng dữ liệu từ client đến database, đồng thời hiểu cách các module tương tác với nhau và với các dịch vụ ngoài.
2. **Hoàn thiện hệ thống xác thực và phân quyền:**
   * Tích hợp thành công Kinde Authentication theo chuẩn OAuth2/OIDC, hỗ trợ đăng nhập qua Google, GitHub và Microsoft.
   * Xây dựng Actor Module cho phép quản lý nhiều loại người dùng (customer, vendor, shipper) và ánh xạ chính xác với thông tin đăng nhập.
3. **Phát triển các tính năng thực tế:**
   * Xây dựng Notification Module và Blogs Workflow với cơ chế kiểm duyệt nội dung (Content Moderation).
   * Triển khai Cart Authentication Middleware để bảo vệ giỏ hàng, đảm bảo tính toàn vẹn và bảo mật.
   * Hoàn thiện Promotion Module với đầy đủ logic áp dụng mã giảm giá và xử lý xung đột khuyến mãi.
4. **Cải thiện kỹ năng kiểm thử và tối ưu hệ thống:**
   * Thực hiện kiểm thử API và workflow bằng Postman và log server.
   * Tối ưu hóa luồng xác thực, caching token, chuẩn hóa phản hồi API và giảm thiểu lỗi hệ thống.
5. **Kỹ năng viết tài liệu và báo cáo kỹ thuật:**
   * Hoàn thiện tài liệu mô tả chi tiết các module, workflow, quy trình tích hợp Kinde và cơ chế bảo mật.
   * Chuẩn bị slide báo cáo tiến độ, tổng hợp kiến thức tuần – tuần và trình bày rõ ràng, logic.

**Nhìn chung**, đợt thực tập không chỉ giúp em củng cố kiến thức về NodeJS, ExpressJS và MedusaJS, mà còn rèn luyện kỹ năng phát triển hệ thống thực tế, giải quyết vấn đề kỹ thuật, quản lý workflow và tích hợp các dịch vụ bên ngoài. Kết quả này tạo nền tảng vững chắc cho việc tiếp tục nghiên cứu nâng cao, mở rộng hệ thống hoặc áp dụng vào các dự án thương mại điện tử trong tương lai.