

# Phát triển và chuyển giao nhanh sản phẩm, dịch vụ tài chính số sử dụng nền tảng low-code

Phan Thanh Đức<sup>1</sup>, Nguyễn Ngọc Quang<sup>2</sup>

<sup>1</sup>Học viện Ngân hàng, Việt Nam <sup>2</sup>Đại học Công Nghiệp Hà Nội, Việt Nam

Ngày nhận: 27/05/2024

Ngày nhận bản sửa: 03/07/2024

Ngày duyệt đăng: 05/07/2024

**Tóm tắt:** Cách mạng công nghiệp lần thứ 4 ảnh hưởng sâu sắc đến đời sống xã hội nói chung cũng như lĩnh vực tài chính nói riêng. Cách mạng công nghiệp lần thứ 4 không chỉ tạo cơ sở cho việc phát triển các mô hình kinh doanh mới, đa dạng kênh phân phối và nâng cao trải nghiệm người dùng mà còn thay đổi phương thức phát triển sản phẩm dịch vụ tài chính trên môi trường số dựa trên những nền tảng và phương pháp phát triển mới. Các nền tảng và phương pháp phát triển mới cho phép rút ngắn thời gian phát triển và chuyển giao sản phẩm dịch vụ tài chính số mà vẫn đảm bảo đáp ứng các yêu cầu bảo mật, thuận tiện cho việc triển khai trên hạ tầng sẵn có cũng như việc chuyển đổi lên môi trường đám mây. Bằng phương pháp tổng quan tài liệu và phân tích, bài báo phân tích sâu về nền tảng low-code và đề xuất phương pháp Scrum- Extreme Programming nhằm phát triển phần mềm sử dụng nền tảng low-code gắn với quy trình tích hợp liên tục- chuyển giao liên tục (Continuous

## Develop and quickly deliver digital financial products and services using low-code platforms

**Abstract:** The 4th Industrial Revolution has a profound impact on social life in general as well as the Finance sector in particular. Industrial Revolution 4.0 not only creates a basis for developing new business models, diversifying distribution channels and improving user experience, but also changes the way of developing financial products and services in the digital environment based on new platforms and development methods. New platforms and development methods allow for shortening the time to develop and transfer financial and banking products and services while still ensuring they meet security requirements and facilitate deployment on the infrastructure as well as conversion to a cloud environment. Using the method of document review and analysis, the article deeply analyzes the low-code platform and proposes combined method Scrum- Extreme Programming for software development using the low-code platform integrated with Continuous Integration- Continuous Delivery process.

**Keywords:** Low-code platform, Scrum- XP, CI/CD

DOI: 10.59276/JELB.2024.07CD.2751

Phan, Thanh Duc<sup>1</sup>, Nguyen, Ngoc Quang<sup>2</sup>

Email: ducpt@hvn.edu.vn<sup>1</sup>, quang@upnext.vn<sup>2</sup>

<sup>1</sup>Banking Academy of Vietnam, <sup>2</sup>Hanoi University of Industry, Vietnam

*Integration- Continuous Delivery).*

**Từ khóa:** *Nền tảng low-code, Scrum- Extreme Programming, CI/CD*

## 1. Giới thiệu

Trong lịch sử phát triển, nền công nghiệp toàn cầu đã trải qua 3 cuộc cách mạng công nghiệp (CMCN) và đang trong quá trình của cuộc CMCN 4.0. CMCN 4.0 với đặc trưng là sự kết hợp các công nghệ của nhiều lĩnh vực, làm mờ ranh giới giữa vật lý, kỹ thuật số và sinh học. CMCN 4.0 được xây dựng và phát triển trên các trụ cột kỹ thuật chính của CMCN 4.0 là: Dữ liệu lớn (Big Data), Trí tuệ nhân tạo (AI) và Vạn vật kết nối (Internet of Thing). So với các cuộc cách mạng trước đây, CMCN 4.0 phát triển theo hàm số mũ chứ không phải là tốc độ tuyến tính. Các ứng dụng điển hình của CMCN 4.0 như trong lĩnh vực phần mềm (ChatGPT, siêu ứng dụng Grab, Zalo...), lĩnh vực y tế (trợ lý ảo IBM Watson, hệ thống PACS...), lĩnh vực sản xuất (nhà máy thông minh,...), lĩnh vực nông nghiệp (trang trại kỹ thuật số...).

CMCN 4.0 đã và đang có nhiều tác động sâu sắc đến sự phát triển nền kinh tế nói chung như sử dụng tư liệu sản xuất mới với những phương thức sản xuất mới trong nền kinh tế số ((Phan, 2023), (Tô, 2022)). Trong nền kinh tế số, dữ liệu đóng vai trò là "tư liệu sản xuất" và phương thức sản xuất dựa trên nguồn tài nguyên đặc biệt này cũng hoàn toàn khác các cuộc CMCN trước đây- hướng tới sự tự động hoá một cách thông minh, không cần sự can thiệp của con người. Và trong lĩnh vực chịu ảnh hưởng sâu sắc và toàn diện của CMCN 4.0- ngành tài chính- ngân hàng, tự động hóa đã trở thành mục tiêu cho việc vận hành các hoạt động của mỗi định chế tài chính, dù là ngân hàng thương mại hay các công ty công

nghệ tài chính Fintech, để có thể tận dụng các thành tựu công nghệ nhằm xây dựng các mô hình kinh doanh số của mình. Các tác động của CMCN đối với ngành tài chính- ngân hàng có thể kể đến là: (1) phát triển, mở rộng và đa dạng hóa kênh phân phối; (2) mở rộng các mô hình phát triển mới; (3) tự động hóa quy trình bằng robot; (4) nâng cao trải nghiệm khách hàng và (5) ứng dụng nền tảng công nghệ mới đẩy nhanh chuyển đổi số trong dịch vụ tài chính.

Trong khi các tác động (1) đến (4) đã được nhiều nghiên cứu đề cập ((Phan, 2023), (Tô, 2022)), việc ứng dụng nền tảng công nghệ mới đẩy nhanh chuyển đổi số trong dịch vụ tài chính đang là vấn đề mới mẻ và chưa có nhiều công trình nghiên cứu có liên quan được công bố. Khác với các dịch vụ sản phẩm tài chính truyền thống, các sản phẩm dịch vụ tài chính số (Digital Financial Products and Services - DFPS) là việc sử dụng nền tảng số để cung cấp các sản phẩm và dịch vụ tài chính trực tiếp tới các thiết bị kỹ thuật số của khách hàng, chẳng hạn như thiết bị cầm tay thông minh hoặc thẻ thanh toán (G20, 2016). Về bản chất, việc phát triển các DFPS chính là việc phát triển các ứng dụng trên nền tảng số, bao gồm từ các nền tảng thiết bị thông minh (mobile apps) hoặc nền tảng web (web apps). Tuy nhiên, việc phát triển ứng dụng trên nền tảng số có những yêu cầu đặc thù như quá trình phát triển liên quan đến nhiều bên tham gia, yêu cầu người dùng thường xuyên thay đổi, các bước triển khai phụ thuộc vào các nền tảng cụ thể, yêu cầu về thời gian phát triển nhanh, yêu cầu bảo mật cao. Trước đây, việc phát triển các DFPS thường được gọi là hoạt động "Tin học hóa sản phẩm

dịch vụ” và đã được nhiều nghiên cứu đề cập tới ((Phan, 2013), (Adedeji, 2020)). Tuy nhiên cùng với sự ra đời của các thành tựu mới trong lĩnh vực công nghệ thông tin như nền tảng low-code/no-code, việc phát triển các DFPS đòi hỏi các cách tiếp cận mới và được kỳ vọng sẽ mang lại những lợi ích mới, đẩy nhanh chuyển đổi số trong dịch vụ tài chính. Theo Finastra (2021), 21% các tổ chức tài chính- ngân hàng được khảo sát cho rằng sẽ tập trung tăng cường đầu tư vào nền tảng công nghệ mới. Các nền tảng công nghệ mới được kỳ vọng cho phép các định chế tài chính nhanh chóng phát triển, triển khai phần mềm nội bộ hoặc cung cấp dịch vụ cho khách hàng. Các đối tác cung cấp các giải pháp cho các định chế tài chính cũng sử dụng các công nghệ mới này để nâng cấp giải pháp và phát triển giải pháp mới. Trong các nền tảng công nghệ mới, nền tảng công nghệ low-code (gọi tắt là nền tảng low-code) được sử dụng rộng rãi với sự tăng trưởng nhanh chóng. Theo Gartner (2022),

thị trường của nền tảng phát triển công nghệ low-code đạt tổng trị giá 26,9 tỷ đô vào năm 2023, tăng 19,6% so với năm 2022.

Dữ liệu Bảng 1 cho thấy nền tảng low-code chiếm thị phần lớn nhất. Gartner dự đoán rằng đến năm 2026, 80% các ứng dụng tài chính- ngân hàng được phát triển trên nền tảng low-code (Gartner, 2022). Như vậy, sử dụng nền tảng low-code để phát triển, chuyển giao ứng dụng nhanh là xu hướng tất yếu.

Để đề xuất một phương pháp phát triển mới nhằm phát huy ưu điểm nền tảng low-code, bài báo sử dụng phương pháp tổng quan và phân tích các kết quả nghiên cứu tiên nhiệm ((Phan, 2023), (Tô, 2022), (Da Cruz, 2021)) và dữ liệu được công bố bởi các công ty lớn, có uy tín lớn trên thế giới ((IBM, 2024), (Gartner, 2022), (OutSystems, 2021, 2022), (Finastra, 2021)). Từ đó, các tác giả đề xuất phương pháp phát triển và chuyển giao nhanh sản phẩm, dịch vụ trong lĩnh vực tài chính- ngân hàng, cụ thể: (1) Về công nghệ: sử dụng nền tảng low-code; (2) Về phương pháp phát triển phần mềm: kết hợp phương pháp Scrum- Extreme Programming tích hợp quy trình CI/CD.

Bài báo tập trung làm rõ phương pháp phát triển các DFPS sử dụng nền tảng low-code và được tổ chức thành 3 phần. Phần đầu là đặt vấn đề về việc phát triển DFPS, phần tiếp theo làm rõ bản chất của nền tảng low-code. Phần cuối là các đề xuất phương pháp Scrum- XP trên nền tảng low-code và các kiến nghị, kết luận có liên quan.

## 2. Nền tảng công nghệ low-code

### 2.1. Lịch sử phát triển và đặc trưng của nền tảng low-code

Hiện nay, chưa có định nghĩa thống nhất chung cho nền tảng low-code. Một trong các định nghĩa được sử dụng rộng rãi do

**Bảng 1. Doanh thu của các nền tảng công nghệ**

Nền tảng công nghệ	2021	2022	2023	2024
Nền tảng ứng dụng Low-Code (LCAP)	6.324	7.968	9.960	12.351
Nền tảng tích hợp PaaS (iPaaS)	4.680	5.668	6.668	7.838
Nền tảng tự động hóa quy trình nghiệp vụ (BPA)	2.416	2.585	2.761	2.940
Nền tảng phát triển đa trải nghiệm (MDXP)	2.081	2.508	2.999	3.563
Tự động hóa quy trình robotic (RPA)	2.350	2.892	3.401	3.879
Nền tảng phát triển và tự động hóa cơ bản (CDAP)	554	732	953	1.232

*Nguồn: Finastra (2021)*

IBM đưa ra: “*Low-code là một phương pháp phát triển phần mềm trực quan cho phép xây dựng và triển khai nhanh nhờ tối thiểu hóa công việc lập trình*” (IBM, 2024). Mô hình nền tảng low-code xuất hiện vào đầu những năm 2000 khi nhà phát triển nhận thấy độ phức tạp, thời gian phát triển của phương pháp phát triển phần mềm truyền thống ngày càng lớn. Trong thời gian này, để xây dựng các phần mềm ứng dụng, yêu cầu các nhà phát triển phải có kỹ năng lập trình chuyên sâu và thông thạo về ngôn ngữ lập trình. Đây là rào cản, hạn chế số lượng nhân sự có thể tham gia phát triển phần mềm.

Nhằm khắc phục hạn chế này, phiên bản đầu tiên của nền tảng low-code hỗ trợ giao diện đồ họa cho phép người dùng kéo-thả (drag-and-drop) cấu phần có sẵn của nền tảng để thiết kế các chức năng theo yêu cầu. Người sử dụng nền tảng low-code không cần có nhiều kinh nghiệm lập trình. Do đó, nhân viên phân tích nghiệp vụ hay quản trị dự án hoàn toàn có thể tham gia phát triển ứng dụng trên nền tảng low-code. Tuy nhiên, tính năng và hiệu năng của phiên bản này tương đối thô sơ so với các tiêu chuẩn ngày nay. Microsoft Sharepoint là một ví dụ về nền tảng low-code cho giai đoạn này. Sử dụng nền tảng Sharepoint, người dùng không cần có kiến thức chuyên sâu về kỹ thuật vẫn có thể tạo và triển khai các website, và ứng dụng web bằng giao diện thiết kế đơn giản.

Với yêu cầu đồng thời giảm thời gian phát triển phần mềm và giảm chi phí, cuối những năm 2010 đánh dấu một bước ngoặt quan trọng đối với các nền tảng low-code. Các nền tảng low-code trở nên phổ biến, dễ tích hợp và có khả năng mở rộng cao, cho phép tổ chức sử dụng chúng để phát triển nhiều ứng dụng hơn. Những ứng dụng chính được xây dựng trên nền tảng low-code trong giai đoạn này: i) Bộ giải pháp

cho doanh nghiệp; ii) Phát triển ứng dụng điện thoại trên đa nền tảng; iii) Ứng dụng AI và tự động hóa.

Đầu năm 2020, các nền tảng low-code tiếp tục phát triển, thích ứng với nhu cầu luôn thay đổi của ngành phát triển phần mềm. Một số xu hướng và sự phát triển chính áp dụng nền tảng low-code hiện nay:

- *Kết hợp cùng với phương pháp phát triển phần mềm truyền thống*: cho phép tổ chức đạt được sự cân bằng giữa tốc độ, khả năng tùy chỉnh, khả năng tích hợp với hệ thống hiện có và dễ dàng mở rộng.

- *Mọi người đều có thể phát triển ứng dụng*: nền tảng low-code trở nên thân thiện, dễ sử dụng cho phép cá nhân không thành thạo về kỹ thuật (citizen developer) có thể tham gia phát triển.

- *Phát triển theo hướng AI (AI-driven)*: AI và học máy đều đã được hỗ trợ trong nền tảng low-code. Do đó, nhà phát triển có thể sử dụng dễ dàng.

- *Cộng tác giữa các thành viên và DevOps*: nền tảng low-code hỗ trợ quy trình DevOps, cho phép cộng tác chặt chẽ giữa nhà phát



Nguồn: Gartner (2022)

**Hình 1. Xếp hạng các nhà cung cấp nền tảng low-code theo đánh giá của Gartner**

triển, nhóm nghiệp vụ và nhóm vận hành.

- *Tuân thủ chính sách và bảo mật*: nền tảng low-code tuân thủ chính sách và yêu cầu bảo mật.

Theo thống kê tính đến năm 2022, các nhà cung cấp nền tảng low-code dẫn đầu thị trường (Gartner, 2022): Mendix, Outsystems, Salesforce, ServiceNow và Microsoft PowerApps. Các đặc trưng của nền tảng low-code bao gồm:

- *Công cụ trực quan tạo mô hình*: cho phép nhà phát triển dễ dàng thiết kế giao diện, luồng dữ liệu, chức năng, giao diện tích hợp bằng cách kéo-thả các cấu phần được định nghĩa trước.
- *Phát triển theo hướng model-based*: cho phép tạo ứng dụng bằng các phương pháp và mô hình trực quan. Từ đó, nhà phát triển dễ dàng hiểu được logic ứng dụng đang

được xây dựng.

- *Mẫu dựng sẵn (Pre-build templates)*: hỗ trợ các mẫu dựng sẵn, tương tự như chợ ứng dụng. Nhà phát triển có thể tìm kiếm, tái sử dụng và tùy chỉnh từ các mẫu dựng sẵn này.
- *Khả năng truy cập đa nền tảng*: các ứng dụng được xây dựng trên nền tảng low-code có thể chạy trên nhiều nền tảng, hệ điều hành khác nhau. Đặc biệt đối với ứng dụng di động, cho phép dễ dàng triển khai trên nền tảng phổ biến: iOS và Android.
- *Bảo mật*: tuân thủ yêu cầu bảo mật của ISO 27000, NIST, CIS, GDPR...
- *Khả năng mở rộng*: cho phép dễ dàng mở rộng hệ thống theo scale-up, scale-out.

## 2.2. So sánh phát triển ứng dụng trên nền tảng low-code và theo mô hình truyền thống

**Bảng 2. So sánh phương pháp phát triển ứng dụng trên nền tảng low-code và phát triển ứng dụng theo mô hình truyền thống**

STT	Tiêu chí	Phát triển ứng dụng trên nền tảng low-code	Phát triển ứng dụng theo mô hình truyền thống
01	Công cụ phát triển	Sử dụng nền tảng low-code do các Hãng cung cấp như Mendix, OutSystems, Microsoft PowerApps...	Sử dụng các ngôn ngữ lập trình như: Java, Python, C#.. hay các nền tảng thông dụng như: Spring Boot, Django, .NET...
02	Kỹ năng lập trình	Kiến thức cơ bản	Kiến thức chuyên sâu về lập trình
03	Thời gian phát triển trung bình	1 tháng/1 ứng dụng	6 tháng- 1 năm/1 ứng dụng
04	Tùy chỉnh	Phụ thuộc vào nền tảng low-code	Cao
05	Agility	Các yêu cầu thay đổi có thể thực hiện nhanh chóng. Các tính năng mới có thể được thêm vào dễ dàng.	Các yêu cầu thay đổi thường tốn nhiều thời gian
06	Triển khai	Thời gian phát triển nhanh vì tối thiểu hóa việc lập trình	Thời gian phát triển lâu hơn vì ứng dụng được phát triển từ đầu
07	Chất lượng	Tích hợp sẵn kiểm thử tự động và trình gỡ lỗi trực tiếp (live-debugging) nên đảm bảo chất lượng ứng dụng	Thực hiện xây dựng kiểm thử tự động. Do đó, việc kiểm tra và gỡ lỗi tốn nhiều thời gian hơn
08	Bảo hành, bảo trì	Dễ dàng bảo hành, bảo trì vì toàn bộ quá trình bảo hành, nâng cấp, xử lý bảo mật là do nền tảng low-code tự xử lý	Thường phải một nhóm nội bộ để thực hiện định kỳ nâng cấp và bảo hành
09	Cộng đồng hỗ trợ kỹ thuật	Ít cộng đồng hỗ trợ kỹ thuật. Phụ thuộc vào sự hỗ trợ của Hãng	Cộng đồng hỗ trợ kỹ thuật nhiều hơn
10	Mẫu dựng sẵn	Rất nhiều mẫu dựng sẵn	Không có. Thường xây dựng từ đầu

11	Khả năng mở rộng	Dễ dàng mở rộng. Việc mở rộng được xử lý bởi nền tảng low-code	Khó mở rộng quy mô và phụ thuộc vào thiết kế riêng biệt của phần mềm
12	Bảo mật	Đáp ứng tiêu chuẩn ISO 2007 và SOC2. Dễ dàng tùy chỉnh đáp ứng tuân thủ các quy định bảo vệ dữ liệu quan trọng khác.	Phụ thuộc hoàn toàn vào thiết kế và nhóm phát triển phần mềm
13	Đa nền tảng	Hỗ trợ đa nền tảng.	Phụ thuộc vào thiết kế, ngôn ngữ lập trình đã lựa chọn
14	Tỷ suất hoàn vốn ROI	Tỷ suất hoàn vốn > 80%	Tỷ suất hoàn vốn 15-17%

Nguồn: Tác giả tổng hợp

### 2.3. Kiến trúc logic của nền tảng low-code

Nền tảng low-code bao gồm các cấu phần:

- **Lớp UI/UX:** cho phép thiết kế giao diện, luồng chức năng bằng công cụ phát triển tích hợp trực quan (Integrated Development Environment- IDE). Công cụ cho phép kéo và thả các cấu phần có sẵn để thiết kế giao diện người dùng, quy trình và mô hình dữ liệu.

- **Lớp logic:** xử lý logic nghiệp vụ của ứng dụng. Lớp này bao gồm các thành phần:

+ **Quản lý vòng đời ứng dụng:** các công cụ phát triển, gỡ lỗi, triển khai và duy trì các

ứng dụng.

+ **Các cấu phần được thiết kế sẵn của nền tảng:** cho phép nhà phát triển có thể sử dụng luôn.

+ **AI:** cho phép xây dựng mô hình AI và sử dụng các thuật toán học máy đã được hỗ trợ sẵn.

+ **DevOps:** cho phép phát triển và triển khai phần mềm theo hướng tiếp cận DevOps.

+ **Tích hợp:** quản lý tích hợp với các nguồn dữ liệu, các dịch vụ được cung cấp bởi bên thứ 3.

- **Lớp dữ liệu:** quản lý cơ sở dữ liệu của nền tảng và các cơ sở dữ liệu của các ứng dụng của nhà phát triển.

- **Lớp hạ tầng:** cho phép triển khai theo các phương án hạ tầng khác nhau:

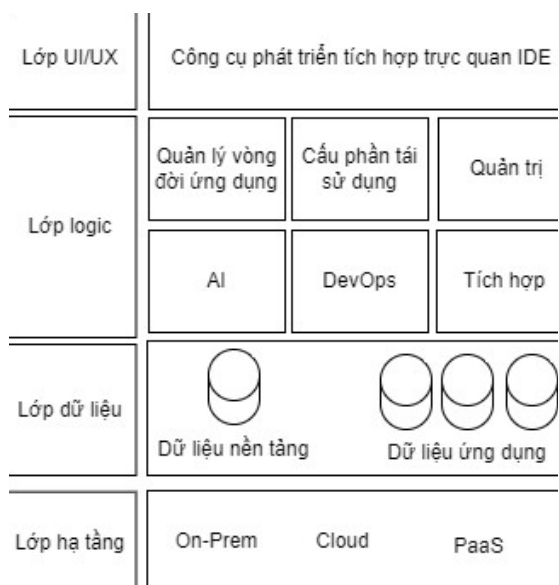
+ **Hạ tầng đám mây.**

+ **Nền tảng dưới dạng dịch vụ (PaaS).**

+ **Triển khai trên hạ tầng sẵn có của khách hàng.**

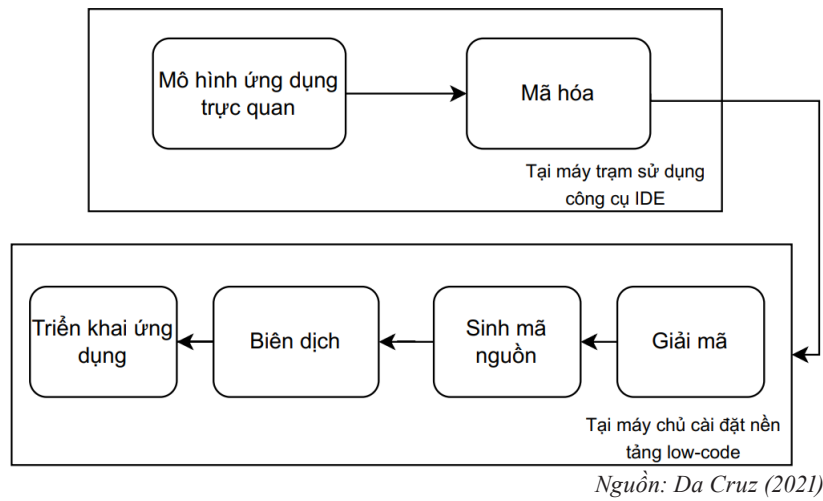
Hình 3 minh họa cách thức nền tảng low-code tự động sinh mã nguồn và triển khai ứng dụng từ bản thiết kế của nhà phát triển thông qua các bước:

**Bước 1 (Tại máy trạm cài đặt công cụ IDE):** nhà phát triển sử dụng công cụ IDE để thiết kế giao diện, quy trình nghiệp vụ và mô hình dữ liệu. Bước này sẽ bao gồm tự động sinh mã, gỡ lỗi, kiểm thử, kiểm soát phiên bản, quản lý sự kiện của ứng dụng... Mỗi nền tảng low-code sẽ tạo ra mã riêng trước khi gửi về máy chủ. Các mã tạo ra thường



Nguồn: Tác giả tổng hợp

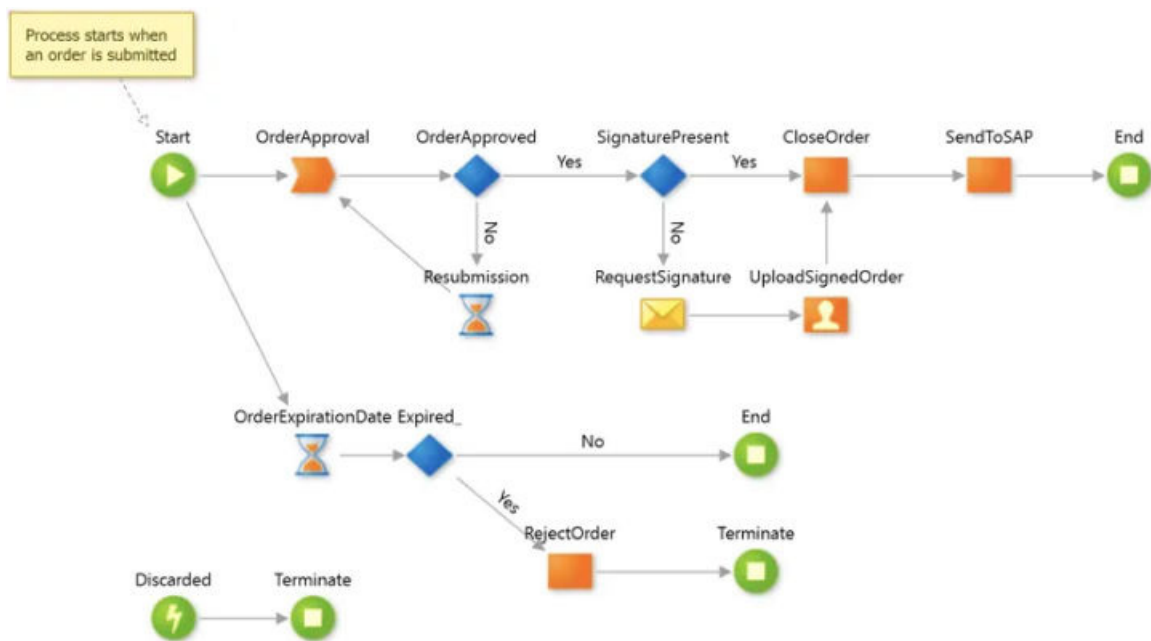
**Hình 2.**  
**Mô hình logic của nền tảng low-code**



**Hình 3. Các bước triển khai ứng dụng trên nền tảng low-code**

có các định dạng như JSON, XML...  
 Bước 2 (Tại máy chủ cài đặt nền tảng low-code): Sau khi nhận được, dữ liệu sẽ được giải mã. Tại bước tiếp theo, nền tảng low-code sẽ tự động sinh mã nguồn tương ứng với ngôn ngữ lập trình được hỗ trợ. Cuối cùng, mã nguồn này sẽ được biên dịch và triển khai trên nền tảng low-code. Để phát triển nhanh các ứng dụng, module

tự động hóa quy trình (Business Process Automation) là một trong tính năng lõi của nền tảng low-code. Tính năng này cho phép nhà phát triển dễ dàng thiết kế các quy trình nghiệp vụ phức tạp với giao diện kéo thả. Module này thường được phát triển trên mô hình quy trình nghiệp vụ BPMN 2.0. Hình 4 minh họa một quy trình đặt hàng được xây dựng trên nền tảng low-code của



Nguồn: OutSystems (2023)

**Hình 4. Giao diện thiết kế quy trình đặt hàng trên nền tảng low-code OutSystems**

hãng OutSystems. Quy trình được bắt đầu khi nhận được một yêu cầu đặt hàng và được thực hiện theo luồng nghiệp vụ mô tả ở mức khái niệm (conceptual) giúp đơn giản hoá việc thiết kế tổng thể lúc ban đầu. Khác với các môi trường phát triển BPM thông thường vốn tập trung vào các thao tác chi tiết, đối với môi trường low-code, ngôn ngữ BPMN được ẩn sau giao diện thiết kế luồng nghiệp vụ trực quan của môi trường IDE. Điều này giúp đẩy nhanh quá trình phát triển và thuận lợi cho việc chuyên môn hoá công việc giữa các nhóm phát triển.

Các nền tảng low-code đã được sử dụng rộng rãi để phát triển các ứng dụng nói chung và ứng dụng tài chính- ngân hàng nói riêng. Chỉ riêng bốn nền tảng low-code dẫn đầu (Gartner, 2022) đã có hơn 5.000 khách hàng, bao gồm các khách hàng lớn như: Intel, Coca-Cola, KPMG, Singtel....

Trong lĩnh vực tài chính – ngân hàng, một số ứng dụng điển hình được phát triển trên nền tảng low-code (OutSystems, 2022), (OutSystems, 2021), (Ranosys, 2023)). Cụ thể:

- Western Union là nhà cung cấp dịch vụ chuyển tiền quốc tế lớn nhất thế giới. Western Union đã sử dụng nền tảng low-code của OutSystems để phát triển và triển khai thế hệ mới ngân hàng số (Next-gen Digital Banking) gồm hơn 20 ứng dụng cung cấp dịch vụ ngân hàng số cho hơn 200 nước trong vòng 11 tháng. Trước khi quyết định sử dụng nền tảng low-code, đội ngũ 10 nhân sự của Western Union đã thực hiện một dự án thiết kế giao diện người dùng để đánh giá hiệu quả. Thời gian của dự án kéo dài 6 tháng để hoàn thành. Sau khi chuyển sang sử dụng nền tảng low-code, thời gian thực hiện cho khối lượng công việc này chỉ mất 15 ngày.

- Edelweiss là một công ty dịch vụ tài chính Ấn Độ với đối tượng khách hàng là

SME. Edelweiss đã sử dụng nền tảng low-code OutSystems để phát triển và ra mắt nền tảng cho vay kỹ thuật số chỉ trong vòng 6 tháng. So với dự toán ban đầu khi dự kiến phát triển theo mô hình truyền thống, Edelweiss đã tiết kiệm 2/3 chi phí và tốc độ nhanh hơn 20%.

- Ngân hàng TMCP Phương Đông OCB đã sử dụng nền tảng low-code để phát triển dịch vụ ngân hàng điện tử Mobile Banking. Với số lượng tính năng lớn, tuy nhiên ứng dụng được phát triển và triển khai trong vòng 3 tháng.

### **3. Đề xuất phương pháp phát triển sản phẩm dịch vụ tài chính số trên nền tảng low-code kết hợp phương pháp Scrum và Extreme Programming (Scrum-XP)**

Với ưu điểm đáp ứng yêu cầu phát triển nhanh sản phẩm, dịch vụ, bài báo lựa chọn low-code là nền tảng kỹ thuật để đề xuất phương pháp phát triển và chuyển giao nhanh sản phẩm, dịch vụ tài chính số.

Trong nền tảng low-code có sẵn thành phần DevOps cho phép phát triển phần mềm theo quy trình tích hợp liên tục - triển khai liên tục (Continuous Integration - Continuous Delivery, thường gọi là quy trình CI/CD). Quy trình CI/CD là một quy trình phát triển phần mềm tập trung cải thiện tốc độ và chất lượng phát triển phần mềm thông qua tự động hóa quy trình phát triển (quy trình CI) và quy trình triển khai ứng dụng (quy trình CD) (Rossel, 2017). Trong đó, quy trình CI thực hiện tự động hóa tích hợp các thay đổi của mã nguồn vào ứng dụng. Cụ thể, khi một nhóm phát triển phần mềm làm việc cùng nhau, quy trình CI đảm bảo các phiên bản mới nhất của mã nguồn được tích hợp tự động vào một kho lưu trữ chung. Bên cạnh đó, quy trình CD là quy trình triển khai tự động các phiên bản mới nhất của ứng dụng vào môi



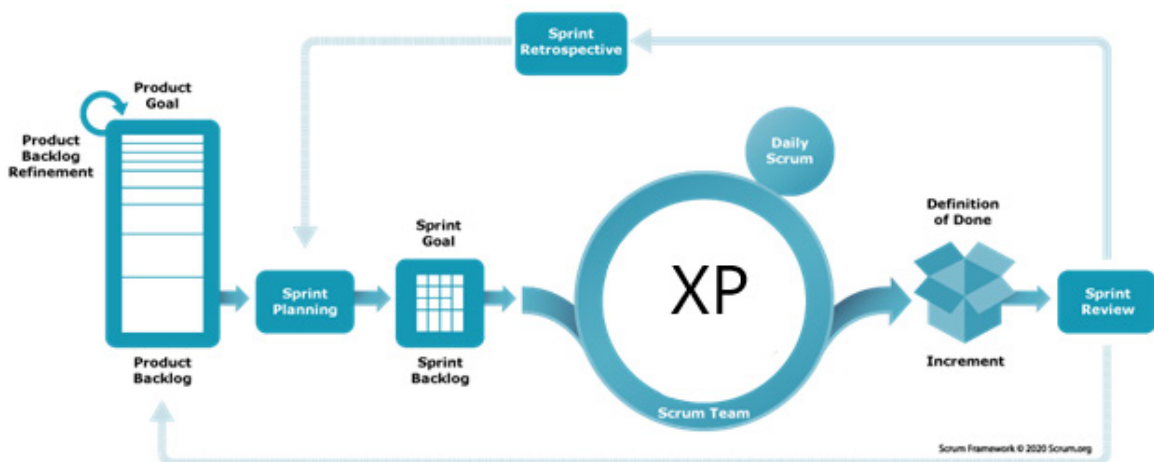
trường kiểm thử hoặc môi trường thật. Do đó, quy trình CD giúp giảm thời gian, công sức triển khai ứng dụng, đồng thời tăng tính nhất quán và độ tinh cậy. Như vậy, với quy trình CI/CD, với mỗi thay đổi trong mã nguồn, mã nguồn sẽ được tự động kiểm tra, tích hợp và triển khai một cách liên tục, giúp tăng tốc độ phát triển và đảm bảo chất lượng phần mềm.

Quy trình CI/CD này là một đặc tính tương đồng của phương pháp phát triển sản phẩm linh hoạt (Agile). Phương pháp phát triển sản phẩm linh hoạt đáp ứng các yêu cầu phát triển nhanh như: tính tương tác cao, coi trọng vai trò khách hàng và khả năng đáp ứng thay đổi nhanh. Một số phương pháp phát triển phần mềm tiêu biểu thuộc lớp các phương pháp phát triển nhanh, bao gồm Extreme Programming (XP), Scrum và Adaptive Software Development (ASD). Trong các phương pháp này, Scrum và ASD là các phương pháp thiên về lĩnh vực quản lý. Scrum đưa ra một quy trình chặt chẽ, trong đó nêu rõ vai trò của những thành viên tham gia dự án cũng như những hoạt động cần phải tiến hành trong quá trình thực hiện dự án. ASD đưa ra một khung quản lý chung, có nhiều tùy chọn cho phép những người quản lý áp dụng linh

hoạt. XP lại tập trung vào các kỹ thuật áp dụng trong lập trình. Dựa trên đặc tính của từng phương pháp, bài báo đề xuất kết hợp phương pháp Scrum và XP, tích hợp quy trình CI/CD trên nền tảng low-code để rút ngắn thời gian phát triển và chuyển giao sản phẩm phần mềm dịch vụ tài chính- ngân hàng mà vẫn đảm bảo đáp ứng các yêu cầu bảo mật, thuận tiện cho việc triển khai trên hạ tầng sẵn có cũng như việc chuyển đổi lên môi trường Cloud.

Phương pháp Scrum-XP đề xuất được minh họa trong Hình 5. Điểm quan trọng trong phát triển phần mềm là xây dựng danh sách các yêu cầu (Product Backlog). Các nền tảng low-code đều có số lượng lớn các ứng dụng, dự án mẫu trên chợ ứng dụng. Do đó, giai đoạn tạo mẫu phần mềm (prototype) sẽ được sử dụng chính trong quá trình thu thập và phân tích yêu cầu của khách hàng. Từ đó, tăng tính tương tác, lấy khách hàng là trung tâm và dễ dàng đáp ứng với yêu cầu thay đổi nhanh.

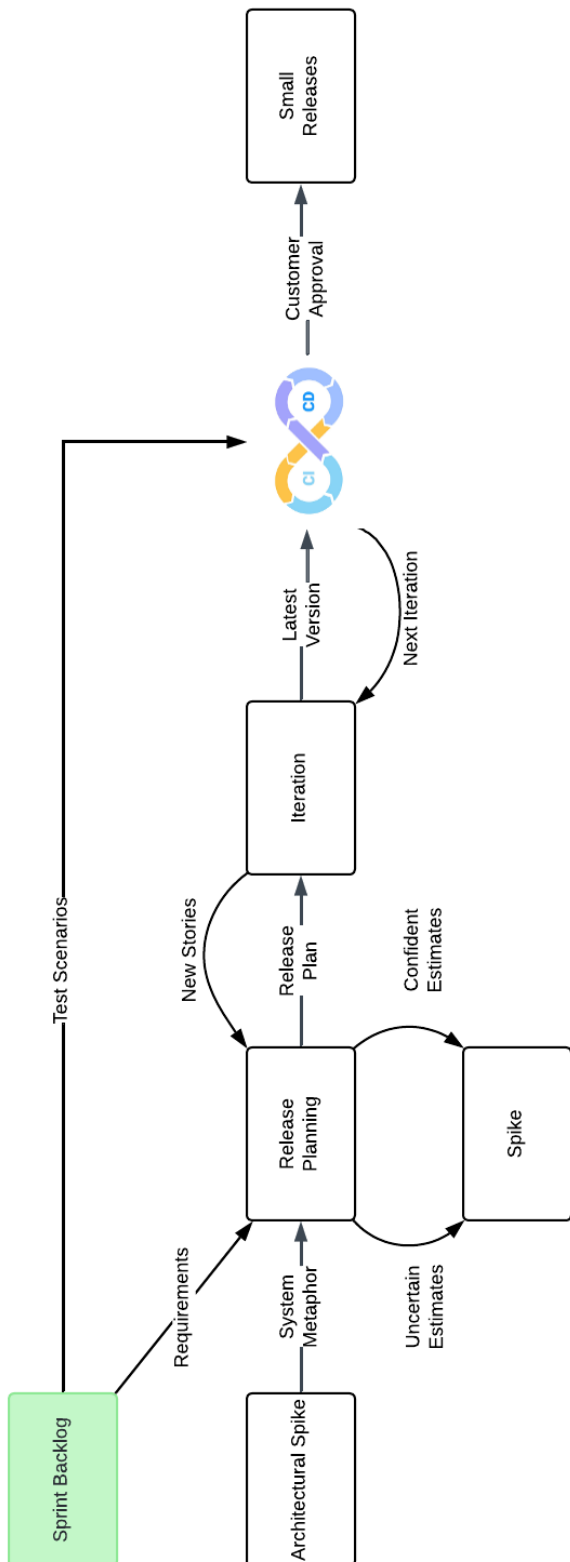
Khác với phương pháp Scrum tiêu chuẩn (Ken Schwaber, 2020), trong phương pháp đề xuất, mỗi Sprint sẽ gồm các bước XP biến thể được minh họa trong Hình 6. Kết quả của mỗi Sprint sẽ là các bản phát hành phần mềm bao gồm các tính năng được



Nguồn: Tác giả tự tổng hợp

Hình 5. Phương pháp Scrum-XP

miêu tả trong danh sách tính năng yêu cầu của Sprint (Sprint Backlog). Các bước XP đề xuất sẽ được thực hiện trong giai đoạn



Nguyên: Tác giả tự tổng hợp

Hình 6. Các bước XP trong phương pháp Scrum-XP đề xuất

này bao gồm:

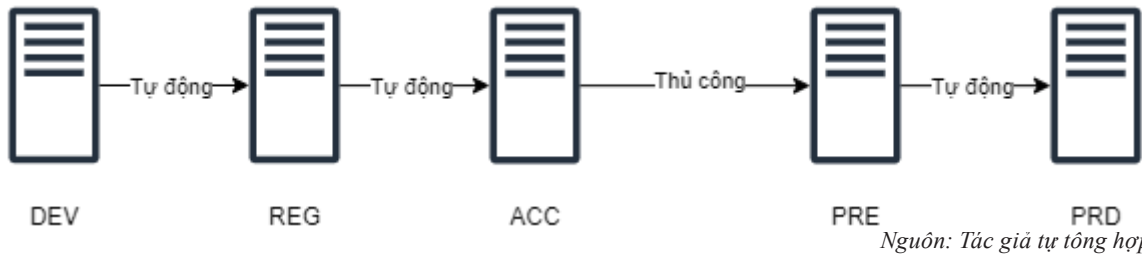
- Phân tích thiết kế: Ấn dụ hệ thống.
- Lập trình: Lập trình đôi.
- Kiểm thử: Phát triển dựa trên thử nghiệm.
- Đảm bảo chất lượng: Tiêu chuẩn mã nguồn, tái cấu trúc và quyền sở hữu mã tập thể.

Bên cạnh đó, bài báo đề xuất việc tích hợp quy trình CI/CD trong phương pháp Scrum-XP. Mặc dù như đã đề cập ở trên, trong các nền tảng low-code có sẵn thành phần DevOps cho phép phát triển phần mềm theo quy trình CI/CD. Tuy nhiên, quy trình CI/CD cần được tinh chỉnh (tuning) để phù hợp với từng phương pháp. Trong phương pháp Scrum-XP đề xuất này, đầu vào của quy trình CI/CD sẽ là các đặc tả câu chuyện người dùng (user stories) và đầu ra là tính năng phần mềm tương ứng đã được miêu tả trong tài liệu kiểm thử chấp nhận của người dùng (user acceptance testing). Các giai đoạn của quy trình CI/CD đề xuất được minh họa trong Hình 7.

Quy trình CI/CD đề xuất bao gồm các giai đoạn:

- *Giai đoạn phát triển ứng dụng (DEV)*: Nhà phát triển sẽ xây dựng ứng dụng và thực hiện kiểm thử đơn vị bằng dữ liệu giả lập.
- *Giai đoạn kiểm thử hồi quy (REG)*: thực hiện chiến lược kiểm thử hồi quy nhằm đảm bảo tính năng mới không ảnh hưởng đến các tính năng hiện có.
- *Giai đoạn kiểm thử chấp nhận (ACC)*: người sử dụng cuối kiểm thử tính đúng đắn, hợp lệ của ứng dụng.
- *Giai đoạn tiền triển khai (PRE)*: thiết lập môi trường tương đồng với môi trường thật để thử nghiệm ứng dụng nhằm phát hiện các vấn đề khi triển khai thực tế.
- *Giai đoạn triển khai (PRD)*: triển khai ứng dụng trên môi trường thực tế.

Hầu hết, các giai đoạn trong quy trình CI/CD đều thực hiện tự động. Sau khi nhà phát triển xây dựng xong tính năng và thực hiện xuất bản thì ứng dụng với tính năng



**Hình 7. Quy trình CI/CD đề xuất**

mới sẽ được kiểm thử hồi quy tự động. Quá trình này đảm bảo tính năng mới không ảnh hưởng đến tính năng hiện có. Tiếp theo, ứng dụng sẽ được kiểm thử chấp nhận. Tại môi trường kiểm thử chấp nhận, người sử dụng cuối có thể lựa chọn phương pháp kiểm thử tự động hoặc kiểm thử thủ công để kiểm tra tính đúng đắn, hợp lệ của tính năng mới. Nếu vượt quá được các bước kiểm thử này, thì ứng dụng đã sẵn sàng để triển khai. Giai đoạn chuyển tiếp ứng dụng từ môi trường kiểm thử sang môi trường tiền triển khai, yêu cầu phải thực hiện trực tiếp bởi thành viên dự án. Việc không cho phép thực hiện tự động chuyển tiếp ở giai đoạn này nhằm ngăn chặn việc lọt ứng dụng chưa đảm bảo chất lượng được đưa vào môi trường triển khai. Trước khi triển khai chính thức trên môi trường thật, ứng dụng sẽ được thử nghiệm trên môi trường PRE. Môi trường PRE là môi trường tương đồng với môi trường PROD và được kiểm thử bởi một nhóm nhỏ người sử dụng cuối. Quá trình triển khai ứng dụng từ môi trường PRE sang môi trường PROD là tự động nếu ứng dụng đã vượt qua được các trường hợp kiểm thử trên môi trường PRE. Đối với định chế tài chính, để có thể bắt đầu áp dụng phương pháp Scrum-XP đề xuất này thì cần thực hiện các bước sau:

- Về quy trình: xây dựng và chuyển đổi sang mô hình sử dụng phương pháp Scrum - XP. Cụ thể:
- + Cam kết của ban điều hành: ban điều hành cam kết chịu trách nhiệm cung cấp

các nguồn lực trong quá trình xây dựng và chuyển đổi mô hình;

- + Trao quyền cho đội nhóm: các thành viên trong nhóm được trao quyền để đưa ra quyết định nhất định mà không phải thông qua các quy trình phê duyệt;
- + Văn hóa hợp tác: nhóm phát triển cần sự hợp tác chặt chẽ nên cần bố trí làm việc tập trung, trong không gian mở;
- + Lập trình đôi: tất cả yêu cầu phải được thực hiện theo mô hình phát triển theo hướng kiểm thử (Test-Driven Development). Một người viết một bài kiểm thử và người kia lập trình để vượt qua bài kiểm thử.
- Về công nghệ: thiết lập các hệ thống phần mềm được tích hợp với nhau trên hạ tầng công nghệ thông tin hiện có:
  - + Hệ thống quản lý dự án theo phương pháp Scrum - XP: quản lý tiến độ công việc, quản lý vấn đề;
  - + Hệ thống low-code: phát triển hạ tầng công nghệ để triển khai nền tảng low-code; Hệ thống DevOps: quản lý quy trình CI/CD.
- Về con người:
  - Tham gia khóa đào tạo về phương pháp Scrum – XP và DevOps.
  - Tự đào tạo thông qua các cổng đào tạo của các nền tảng low-code.

#### 4. Kết luận

Với các đặc trưng riêng có là tài nguyên là dữ liệu số và nền tảng công nghệ số, nền kinh tế số đòi hỏi phương thức sản xuất

mới- tự động hoá. Trong các nền tảng công nghệ số, low-code là nền tảng công nghệ cho phép phát triển và chuyển giao sản phẩm nhanh mà vẫn tuân thủ các yêu cầu bảo mật, thuận tiện cho việc triển khai trên hạ tầng sẵn có cũng như việc chuyển đổi lên môi trường đám mây. Để hình thành phương pháp phát triển và chuyển giao nhanh sản phẩm, dịch vụ trong bối cảnh chuyển đổi số ngành tài chính- ngân hàng đang diễn ra mạnh mẽ, bài báo đề xuất phương pháp phát triển và chuyển giao nhanh sản phẩm, dịch vụ thông qua kết hợp giữa nền tảng kỹ thuật low-code với các ưu điểm của phương pháp Scrum- XP. Việc kết hợp này là cơ sở cho việc tự động hoá quá trình phát triển và chuyển giao liên tục các sản phẩm, dịch vụ tài chính số- phương thức sản xuất mới trong nền kinh tế số. ■

### Tài liệu tham khảo

- Adedeji, E.A. (2020). *Effect of Computerization on Banks' Performance in Quoted Nigerian Banking Sector*. *Research Journal of Finance and Accounting*, 11(16). <https://doi.org/10.7176/rjfa/11-16-17>
- Da Cruz, M. A., de Paula, H. T., Caputo, B. P., Mafra, S. B., Lorenz, P., & Rodrigues, J. J. (2021). *OLP—A restful open low-code platform*. *Future Internet*, 13(10):249. <https://doi.org/10.3390/fi13100249>.
- Finastra, (2021). *Financial services state of the nation survey 2021*. <https://www.finastra.com/sites/default/files/documents/2021/06/financial-services-state-of-the-nation-survey-2021.pdf> truy cập ngày 26/04/2024.
- GPFI (2016). *High-level principles for digital financial inclusion*. <https://www.gpfi.org/sites/gpfi/files/G20%20High%20Level%20Principles%20for%20Digital%20Financial%20Inclusion.pdf> truy cập ngày 28/06/2024.
- Gartner. (2022). *Forecasts worldwide low-code development technologies market to grow 20% in 2023*. <https://www.gartner.com/en/newsroom/press-releases/2022-12-13-gartner-forecasts-worldwide-low-code-development-technologies-market-to-grow-20-percent-in-2023> truy cập ngày 26/04/2024.
- IBM. (2024). *What is low-code?..* [https://www.ibm.com/topics/low-code#:~:text=for truy cập ngày 26/04/2024.](https://www.ibm.com/topics/low-code#:~:text=for%20truy%20c%E1%BA%A1p%20ng%C3%A0y%2026/04/2024)
- Ken, S. (2020). *What is Scrum?.* <https://www.scrum.org/resources/what-scrum-module> truy cập ngày 26/04/2024.
- OutSystems (2021). *Financial services firm Edelweiss launches digital lending platform* <https://www.outsystems.com/case-studies/edelweiss-loan-origination> truy cập ngày 26/04/2024.
- OutSystems (2022). *Western Union launches next-gen digital banking services in 11 months*. <https://www.outsystems.com/case-studies/western-union-digital-banking-transformation> truy cập ngày 26/04/2024.
- OutSystems (2023). *How to build low-code workflows*. <https://www.outsystems.com/blog/posts/low-code-workflow> truy cập ngày 26/04/2024.
- Phan, T.D. (2013). *Thực trạng và giải pháp quy trình tin học hoá sản phẩm, dịch vụ tại các ngân hàng thương mại Việt Nam*. *Tạp chí Khoa học & Đào tạo Ngân hàng*, 138, 51-53.
- Phan, T.D. (2023). *Từ dữ liệu đến tự động hóa- các trụ cột của nền kinh tế số. Kỳ yếu Hội thảo khoa học quốc gia "Chuyển đổi số: Từ dữ liệu đến tự động hoá"*. *Học viện Ngân hàng, Hà Nội*, 2-10.
- Ranosys, T.L. (2023). *Developing an OutSystems banking application for OCB to deliver better banking experiences*. *Ranosys*. <https://www.ranosys.com/sg/success-stories/case-studies/ocb> ngày 26/04/2024.
- Rossel, S. (2017). *Continuous integration, delivery, and deployment*. Packt Publishing.
- Tô, T.D.L. (2022). *Tác động của Cách mạng công nghiệp lần thứ tư đến hoạt động ngân hàng và xu hướng phát triển ngân hàng ứng dụng công nghệ 4.0*. *Tạp chí Ngân hàng*. <https://tapchinganhang.gov.vn/tac-dong-cua-cach-mang-cong-nghiep-lan-thu-tu-den-hoat-dong-ngan-hang-va-xu-huong-phat-trien-ngan-ha.htm> ngày 26/04/2024.