Giới thiệu chủ đề nội dung thi phần mềm nguồn mở - OLP 2024

1. Tổng quan về LCDP

1.1 LCDP là gì?

Trong kỉ nguyên chuyển đổi số, các tổ chức doanh nghiệp luôn đòi hỏi về tốc độ triển khai nhanh và tính sẵn sàng của các ứng dụng phục vụ các mục tiêu kinh doanh liên tục biến động. Ngày càng nhiều các tổ chức, doanh nghiệp hướng tới sử dụng các nền tảng phát triển ứng dụng để chuẩn hóa về mặt công nghệ, đồng thời đáp ứng được các yêu cầu nâng cao. Các nền tảng phát triển ứng dụng dùng ít mã nguồn đã ra đời để đáp ứng được các xu hướng chuyển đổi công nghệ phù hợp với yêu cầu của thị trường.

Khái niệm nền tảng phát triển dùng ít mã nguồn LCDP (Low-Code Development Platform) lần đầu tiên được đưa ra bởi Forrester vào năm 2014. Nền tảng phát triển dùng ít mã nguồn cung cấp các công cụ giúp đơn giản hóa và đẩy nhanh quá trình phát triển ứng dụng bằng cách giảm thiểu lượng mã hóa thủ công cần thiết. Thay vì viết từng dòng mã, người dùng có thể sử dụng các công cụ trực quan như giao diện kéo thả, mô hình hóa và các thành phần dựng sẵn để xây dựng các ứng dụng chức năng.

1.2 Đối tượng sử dụng và lợi ích

Nền tảng phát triển dùng ít mã nguồn phù hợp với nhiều đối tượng khác nhau, bao gồm:

- Nhà phát triển: LCDP có thể giúp các nhà phát triển làm việc hiệu quả hơn và tạo ra các ứng dụng nhanh hơn.
- Doanh nghiệp: LCDP có thể giúp các doanh nghiệp giảm chi phí phát triển ứng dụng và nhanh chóng thích ứng với những thay đổi trong nhu cầu kinh doanh.
- Người dùng không có kỹ năng lập trình: LCDP cho phép những người không có kỹ năng lập trình cũng có thể tham gia vào quá trình phát triển ứng dụng.

Lợi ích của nền tảng phát triển dùng ít mã nguồn:

- Phát triển nhanh chóng: LCDP giúp rút ngắn thời gian phát triển ứng dụng bằng cách tự động hóa các tác vụ lặp đi lặp lại và cung cấp các công cụ trực quan để thiết kế và xây dựng ứng dụng.
- Giảm chi phí: LCDP có thể giúp giảm chi phí phát triển ứng dụng bằng cách giảm nhu cầu về nguồn nhân lực có trình độ cao và các công cụ phát triển đắt tiền.

- Cải thiện sự linh hoạt: LCDP giúp các doanh nghiệp dễ dàng thích ứng với những thay đổi trong nhu cầu kinh doanh bằng cách cho phép họ nhanh chóng tạo và triển khai các ứng dụng mới.
- Trao quyền cho người dùng: LCDP cho phép những người không có kỹ năng lập trình cũng có thể tham gia vào quá trình phát triển ứng dụng, giúp thúc đẩy sự đổi mới và cộng tác.

1.3 Thị trường cung cấp LCDP

Theo Gartner, thị trường LCDP đang phát triển nhanh chóng với tốc độ tăng trưởng dự kiến đạt 48,2% trong năm 2023. Gartner dự đoán rằng đến năm 2025, 70% các ứng dụng mới sẽ được xây dựng bằng các công cụ phát triển không cần mã hoặc ít mã. Gartner đã đánh giá các nhà cung cấp LCDP hàng đầu trong Magic Quadrant cho Nền tảng Phát triển Đa trải nghiệm (MXDP) năm 2023. Báo cáo xếp hạng các nhà cung cấp dựa trên khả năng thực hiện và tầm nhìn chiến lược. Dưới đây là một số nhà cung cấp LCDP hàng đầu được Gartner công nhận:

- Nhóm dẫn đầu (Leaders): Mendix, OutSystems, Microsoft Power Apps, ServiceNow, Appian, Salesforce
- Nhóm thách thức (Challengers): Oracle, Retool, HCL Software, Zoho
- Nhóm tầm nhìn (Visionaries): Creatio, PegaSystem, Globant (GeneXus)
- Nhóm tiểu ngạch (Niche Players): Huawei, Newgen, Kintone, Unqork

Gartner cũng xác định một số xu hướng chính trong thị trường LCDP, bao gồm:

- Sự gia tăng nhu cầu về các giải pháp toàn diện: Các doanh nghiệp đang tìm kiếm các giải pháp LCDP có thể cung cấp một bộ đầy đủ các tính năng, bao gồm phát triển ứng dụng web, di động, backend và quy trình làm việc.
- Tập trung vào trải nghiệm người dùng: Các nhà cung cấp LCDP đang đầu tư vào việc cải thiện trải nghiệm người dùng của các công cụ của họ để làm cho chúng dễ sử dụng và trực quan hơn.
- Sự tích hợp với các công nghệ mới: Các nền tảng LCDP đang được tích hợp với các công nghệ mới như trí tuệ nhân tạo (AI), học máy (ML) và Internet vạn vật (IoT) để cung cấp các tính năng và khả năng tiên tiến hơn.

Nhìn chung, Gartner tin rằng LCDP là một công cụ mạnh mẽ có thể giúp các doanh nghiệp giảm chi phí phát triển ứng dụng, tăng tốc độ đổi mới và cải thiện hiệu quả hoạt động.

2. Phương pháp xây dựng LCDP

2.1 Mô hình kiến trúc, tính năng của LCDP

Mỗi nhà cung cấp khác nhau đều xây dựng một kiến trúc riêng cho sản phẩm LCDP của mình. Tuy nhiên có thể tổng quát hóa một kiến trúc chung cho LCDP bao gồm 04 tầng như sau:

- Tầng ứng dụng: Cung cấp môi trường đồ họa bao gồm các công cụ và đối tượng giao diện người dùng.

- Tầng tích hợp dịch vụ: Cung cấp các dịch vụ tích hợp với các dịch vụ khác nhau như API, dịch vụ định danh.
- Tầng tích hợp dữ liệu: Cung cấp khả năng tích hợp, thao tác với dữ liệu. Dữ liệu có thể đến từ các nguồn khác nhau.
- Tầng triển khai: Cung cấp triển khai các ứng dụng được phát triển trên các môi trường khác nhau như môi trường đám mây, môi trường server riêng biệt. Khả năng container hóa cũng được xem xét.

Các nền tảng phát triển khác nhau, có thể cung cấp các tính năng khác nhau. Bảng sau tổng hợp chi tiết các tính năng được hỗ trợ bởi các nền tảng phát triển ứng dụng dùng ít mã nguồn.

Tính năng mức cao	**Các tính năng chi tiết**	**Mô tả tính năng**
Các công cụ phát triển trực quan	Công cụ thiết kế trực quan	Công cụ mô hình hóa trực quan được sử dụng để phát triển phần mềm. Thông qua đó, tất cả các thành phần của phần mềm được phát triển bằng cách lựa chọn, sắp xếp, cấu hình và kết nối các thành phần
	Sử dụng kéo thả	Tính năng kéo thả cho phép lập trình viên tạo ứng dụng bằng cách kéo thả các widget tới vị trí thích hợp trên giao diện
	Môi trường phát triển tích hợp	Môi trường phát triển tích hợp cho lập trình viên trong toàn bộ chu trình phát triển
Hỗ trợ tái sử dụng	Các thành phần template được định nghĩa trước	Các thành phần định nghĩa trước bao gồm các widgets, các dịch vụ, bộ kết nối connectors, và các templates có thể được sử dụng trong môi trường phát triển
	Các workflow được xây dựng trước	Cung cấp các workflow phổ biến có thể sử dụng trong phát triển ứng dụng
Đặc tả nguồn và cấu trúc dữ liệu	Đặc tả cấu trúc dữ liệu	Thành phần cho phép đặc tả cấu trúc dữ liệu sử dụng công cụ mô hình hóa trực quan
	Nguồn dữ liệu	Khả năng truy cập tới các nguồn dữ liệu bên ngoài

Hỗ trợ liên thông hệ thống	Kết nối với các nguồn dữ liệu khác nhau	Hỗ trợ kết nối ứng dụng tới các nguồn dữ liệu đa dạng: CSDL quan hệ, CSDL phi quan hệ, hoặc các tệp tin
	Liên thông với các dịch vụ ngoài	Cung cấp chức năng sẵn sàng kết nối với các dịch vụ bên ngoài thông qua các API (RESTAPI, GRAPH/QL)
	Tích hợp với các dịch vụ AI bên ngoài	Cho phép sử dụng các dịch vụ AI chuyên biệt hóa bên ngoài
Đặc tả luồng logic nghiệp vụ	Tạo các quy tắc nghiệp vụ	Đặc tính cho phép thiết lập các luật logic nghiệp vụ
	Soạn thảo luồng nghiệp vụ trực quan	Đặc tính cho phép đặc tả các luật logic nghiệm vụ sử dụng công cụ trực quan
Hỗ trợ kiểm thử	Kiểm thử tự động	Cho phép sinh các test tự động
	Tích hợp với các nền tảng kiểm thử	Kỹ thuật cho phép tích hợp với các nền tảng kiểm thử bên ngoài
Hỗ trợ triển khai	Hỗ trợ triển khai tự động	Công cụ tự động triển khai ứng dụng
	Triển khai trên đám mây	Tính năng cho phép triển khai ứng dụng trên hạ tầng tính toán đám mây
	Triển khai trên hạ tầng cục bộ	Tính năng cho phép triển khai ứng dụng trên hạ tầng tính toán của tổ chức

2.2 Các phương pháp phát triển ứng dụng sử dụng ít mã

i. Phát triển ứng dụng hướng thành phần

Cách phát triển ứng dụng theo phương pháp hướng thành phần CDD (Component-Driven Development) thường dùng để xây dựng ứng dụng có giao diện đồ họa (GUI). CDD là phương pháp phát triển dựa trên việc chia nhỏ giao diện ứng dụng thành các thành phần nhỏ, độc lập và có thể tái sử dụng. Các thành phần này được thiết kế với giao diện xác định rõ ràng, cho phép chúng được kết hợp với nhau để tạo ra các ứng dụng phức tạp.

Phương pháp phát triển ứng dụng này không phải là mới. Nó đã được áp dụng nhiều trong các ngôn ngữ lập trình ứng dụng desktop. Đặc biệt bằng cách sử dụng công cụ kéo thả các thành phần đồ họa giao diện, người lập trình có thể tạo dựng ứng dụng một cách nhanh chóng mà không cần phải viết mã nguồn từ đầu. Đây cũng chính là cách mà các nền tảng LCDP

ngày nay sử dụng để cho phép người dùng phát triển các ứng dụng bằng tính năng kéo thả giao diện trên web.

Phương pháp CDD sẽ chủ yếu tập trung vào tầng ứng dụng trong kiến trúc LCDP. Ưu điểm của phương pháp này là người dùng có thể tạo giao diện ứng dụng nhanh chóng và linh hoạt với nhiều loại nguồn dữ liệu khác nhau thông qua việc sử dụng các bộ kết nối dữ liệu (datasource connectors). Các thành phần đồ họa trong thư viện có thể được tái sử dụng trong nhiều ứng dụng khác nhau, giúp tiết kiệm thời gian và công sức phát triển. Các ứng dụng có thể dễ dàng mở rộng bằng cách thêm các thành phần mới hoặc sửa đổi các thành phần hiện có. Người dùng có thể nhanh chóng thay đổi và thích ứng giao diện của ứng dụng với các yêu cầu mới phát sinh. Mỗi thành phần có thể được tối ưu hóa cho các nhiệm vụ cụ thể, giúp cải thiện hiệu suất tổng thể của ứng dụng.

Ngày nay, ứng dụng web được sử dụng phổ biến trong các hệ thống thông tin. Các ngôn ngữ lập trình frontend theo phương pháp hướng thành phần có thể kể đến như React, Angular, Vue.js, Elm, Svelte. Các nền tảng LCDP thường được xây dựng trên nền móng của một ngôn ngữ phát triển frontend hướng thành phần để cung cấp tính năng tạo ứng dụng bằng công cụ trực quan.

ii. Phát triển ứng dụng hướng mô hình

Cách phát triển ứng dụng hướng mô hình MDD (Model-Driven Development) xuất hiện cùng với sự ra đời của các khung hỗ trợ (framework) phát triển phần mềm. MDD là phương pháp phát triển tập trung vào việc xây dựng các mô hình trừu tượng của hệ thống phần mềm trước khi viết mã. Các mô hình này mô tả các chức năng, hành vi và dữ liệu của hệ thống một cách độc lập với ngôn ngữ lập trình cụ thể. Sau khi các mô hình được thiết lập, mã nguồn có thể được tự động tạo ra từ các mô hình đó, hoặc các nhà phát triển có thể sử dụng các mô hình làm hướng dẫn để viết mã thủ công.

Phương pháp MDD cũng được áp dụng trong các nền tảng LCDP. Nó cho phép người dùng tập trung vào việc xây dựng mô hình dữ liệu của ứng dụng trước khi thiết kế giao diện người dùng (UI). Mô hình dữ liệu nhằm xác định các loại dữ liệu mà ứng dụng sẽ lưu trữ và mối quan hệ giữa chúng. Sau khi mô hình dữ liệu được thiết lập, UI có thể được tạo tự động dựa trên thiết kế của nó.

Khác với phương pháp CDD, MDD giúp tăng năng suất phát triển bằng cách tự động hóa và giảm thiểu sự lặp lại ở các tác vụ chủ yếu ở tầng dịch vụ và dữ liệu của ứng dụng. Ưu điểm của phương pháp này là cho phép biểu diễn một cách tường minh và trực quan kiến trúc nghiệp vụ, mô hình dữ liệu của hệ thống. Chúng giúp đảm bảo rằng các yêu cầu về nghiệp vụ, dữ liệu được hiểu và thực hiện chính xác, từ đó cải thiện chất lượng và tính duy trì liên tục của phần mềm để thích ứng với mỗi sự thay đổi về nghiệp vụ trong tương lai. Tuy nhiên MDD có hạn chế là giao diện (UI) của ứng dụng tạo ra sẽ phải tuân theo một số khuôn mẫu nhất định. Nó thường không đủ linh hoạt để tạo ra các giao diện có sự tương tác phức tạp với người dùng. Do vậy một nền tảng LDCP có thể kết hợp giữa MDD với phương pháp CDD để cho phép thiết kế linh hoạt đồng thời cả mô hình nghiệp vụ và giao diện ứng dụng.

Trong thiết kế các ứng dụng web hiện đại ngày nay, kiến trúc microservice thường được sử dụng để phân tách ứng dụng thành các thành phần backend và frontend có thể hoạt động độc lập. Các nền tảng LCDP hiện đại sẽ thường áp dụng phương pháp MDD để xây dựng thành phần backend và phương pháp CDD cho thành phần frontend. Dựa trên mô hình dữ liệu được thiết kế theo phương pháp MDD, nền tảng LCDP có thể tự động tạo sinh mã nguồn theo một khuôn mẫu cho cả dịch vụ backend và giao diện frontend. Tuy nhiên người dùng sau đó có thể tiếp tục dùng phương pháp CDD để chỉnh sửa, thích ứng giao diện với yêu cầu của nghiệp vụ ứng dụng.

Đối tượng thực hiện mô hình hóa theo phương pháp MDD bao gồm cả dữ liệu và quy trình nghiệp vụ. Có hai cách tiếp cận để xây dựng công cụ hỗ trợ mô hình hóa trong các nền tảng LCDP như sau.

- Ngôn ngữ lập trình chuyên dụng DSLs (Domain-Specific Languages) dùng để thiết kế, mô tả các khái niệm và quy trình trong một lĩnh vực cụ thể. Đây là một dạng ngôn ngữ theo hình thức khai báo (declarative) với cú pháp dành riêng cho lập trình viên trong mô tả cấu trúc dữ liệu và luồng quy trình nghiệp vụ. Tuy nhiên, một giao diện (UI) có thể được xây dựng để hỗ trợ người dùng thực hiện khai báo mô hình theo dạng biểu mẫu (form) thay vì phải viết mã trực tiếp bằng ngôn ngữ DSL. Hầu hết các nền tảng LDCP nằm trong nhóm dẫn đầu theo đánh giá của Gartner đều sở hữu cho mình một ngôn ngữ DSL riêng để phục vụ người dùng phát triển ứng dụng.
- Ngôn ngữ hình thức biểu diễn đồ họa là cách tiếp cận mô hình hóa trực quan bằng hình vẽ. Người dùng có thể kéo thả và khai báo các đối tượng mô tả các khái niệm và quy trình xử lý nghiệp vụ trong hệ thống phần mềm. Các ngôn ngữ mô hình hóa trực quan thường được áp dụng theo tiêu chuẩn, ví dụ như UML (Unified Modeling Language) là ngôn ngữ mô hình hóa hướng đối tượng được dùng phổ biến để mô tả cấu trúc, hành vi và tương tác trong các hệ thống thông tin; BPMN (Business Process Model and Notation) là ngôn ngữ mô hình hóa nghiệp vụ dùng để thể hiện trực quan các bước, hoạt động, quyết định, luồng dữ liệu và các yếu tố khác trong các quy trình quản lý.

iii. Phát triển ứng dụng hướng sự kiện

Cách phát triển ứng dụng hướng sự kiện EDD (Event-Driven Development) xuất hiện cùng với sự ra đời của các mô hình phát triển ứng dụng phân tán. EDD là phương pháp phát triển dựa trên việc chia nhỏ ứng dụng thành các thành phần nhỏ, độc lập và có thể giao tiếp với nhau thông qua các sự kiện. Khi một sự kiện xảy ra, nó sẽ kích hoạt một chuỗi các hành động trong các thành phần khác. EDD giúp xây dựng các ứng dụng linh hoạt, có thể mở rộng và dễ dàng bảo trì.

Phương pháp EDD phù hợp để áp dụng trong các nền tảng LCDP để hỗ trợ xây dựng ứng dụng theo kiến trúc microservice. Khi đó ứng dụng được chia thành nhiều vi (micro) dịch vụ và có thể trao đổi tương tác với nhau qua các sự kiện. Ưu điểm của giải pháp này là các ứng dụng có thể dễ dàng thích ứng với những thay đổi trong yêu cầu vì các thành phần có thể được thêm, sửa đổi hoặc gỡ bỏ mà không ảnh hưởng đến các thành phần khác. Có thể dễ dàng mở rộng bằng cách thêm các thành phần mới để xử lý thêm sự kiện hoặc tăng lưu lượng truy cập. Hệ thống cũng có khả năng phục hồi cao hơn và cải thiện hiệu suất vì tính độc lập của các

thành phần khi chạy. Các sự kiện sẽ chỉ kích hoạt các thành phần cần thiết, tránh lãng phí tài nguyên.

Đặc biệt, các hệ thống có quy trình xử lý tự động hóa theo thời gian thực rất thích hợp để xây dựng bằng phương pháp EDD. Theo đó các quy trình tự động được mô hình hóa dưới dạng các bộ luật E-C-A để thể hiện nghiệp vụ khi một sự kiện (E) xảy ra, với điều kiện (C) được thỏa mãn thì thực hiện thao tác (A). Các loại sự kiện được kích hoạt sẽ bao gồm thời gian (theo chu kỳ), trigger CSDL, thông điệp truyền tin, thư điện tử,...

Ngày nay, nhiều công cụ truyền tin có thể được áp dụng trong phát triển ứng dụng. Chúng được phân loại theo các nhóm gồm: Message Broker (vd., Kafka, RabbitMQ, ActiveMQ) được sử dụng làm bộ trung chuyển có chức năng lưu trữ và gửi tin nhắn theo hàng đợi trong các hệ phân tán; WebSocket sử dụng kết nối TCP song công để truyền dữ liệu thời gian thực giữa máy khách và máy chủ; Giao thức HTTP cho phép kích hoạt các sự kiện một cách chủ động thông qua các điểm truy cập URL hoặc kết nối API. Ngoài ra còn có các công cụ chuyên dụng khác phục vụ cơ chế truyền tin phân tán cho các ứng dụng di động chạy trên các nền tảng Android và IOS (vd., FCM).

iv. Phát triển ứng dụng với sự trợ giúp của AI

Sự phát triển của công nghệ trí tuệ nhân tạo tạo sinh (Generative AI) đã tạo ra một cuộc cách mạng cho phép máy móc làm việc thay thế con người trong một số công việc sử dụng trí óc. Công nghệ này đã được minh chứng là có hiệu quả trong hỗ trợ lập trình máy tính. Do vậy các nền tảng LCDP tiên tiến thường được bổ sung thêm công nghệ AI vào việc hỗ trợ người dùng trong quy trình phát triển ứng dụng để có thể tự động hóa các tác vụ, cải thiện hiệu quả và chất lượng phần mềm tạo ra.

Lợi ích của việc áp dụng AI trong nền tảng LCDP là giải phóng thời gian cho các nhà phát triển để tập trung vào các công việc sáng tạo và chiến lược hơn. AI có thể giúp phát triển phần mềm nhanh hơn và hiệu quả hơn bằng cách tự động hóa các tác vụ và tối ưu hóa quy trình. AI có thể giúp phát triển phần mềm chất lượng cao hơn bằng cách phát hiện lỗi sớm hơn, cải thiện khả năng kiểm thử và đảm bảo tuân thủ các tiêu chuẩn mã hóa. Đồng thời, AI có thể giúp phát triển các giải pháp phần mềm sáng tạo và đổi mới bằng cách phân tích dữ liệu, xác định xu hướng và đưa ra đề xuất.

3. Các giải pháp LCDP dùng phần mềm nguồn mở

Tất cả các sản phẩm nền tảng LCDP đã được Gartner đánh giá trong Magic Quadrant hiện nay đều là phần mềm thương mại. Cũng tương tự như nhiều sản phẩm CNTT truyền thống khác, các phần mềm thương mại có sự tiếp cận, chiếm lĩnh thị trường thường nhanh hơn các phần mềm nguồn mở. Tuy nhiên việc phổ cập các phần mềm thương mại sẽ khó khăn hơn so với phần mềm nguồn mở bởi các lí do về chi phí và sự phụ thuộc, trói buộc vào công nghệ của nhà cung cấp (vendor lock-in).

Gần đây, nhiều nền tảng LCDP dựa trên phần mềm nguồn mở đã được xây dựng để đáp ứng nhu cầu phát triển không ngừng của xã hội. Chúng ta có thể phân loại các nền tảng LCDP dùng mã nguồn mở vào hai nhánh chính như dưới đây.

3.1 Các nền tảng LCDP đa dụng

Các nền tảng LDCP đa dụng được thiết kế để hỗ trợ người dùng phát triển ứng dụng không bị hạn chế trong một miền lĩnh vực cụ thể nào. Chúng thường cung cấp đủ bộ công cụ để nhà phát triển có thể tạo ra được một ứng dụng hoàn chỉnh bao gồm CSDL và giao diện ứng dụng dành cho người dùng. Sau đây là một danh sách khảo sát các nền tảng LCDP đa dụng nguồn mở hiện có.

Budibase

Budibase là một nền tảng mã nguồn mở hỗ trợ người dùng xây dựng các ứng dụng sử dụng nội bộ một cách nhanh chóng. Nó cung cấp ba bộ công cụ hỗ trợ phát triển bao gồm tạo dựng CSDL, thiết kế giao diện ứng dụng, và tự động hóa quy trình làm việc. Có thể kết nối ứng dụng được tạo ra bằng Budibase với các nguồn CSDL hoặc ứng dụng bên ngoài (vd., Google Sheets, Salesforce, Stripe) thông qua truy vấn SQL hoặc kết nối API. Người dùng có thể linh hoạt lựa chọn triển khai ứng dụng sử dụng dịch vụ hạ tầng đám mây hoặc trên máy chủ dùng riêng.

Budibase đi theo cách tiếp cận sử dụng phương pháp hướng thành phần (CDD) để xây dựng các ứng dụng. Do đó người dùng có thể tùy biến, bổ sung thêm các thành phần đồ họa và kết nối dữ liệu dưới dạng các plugins của nền tảng. Kiến trúc phần mềm của Budibase đã phân tách được giữa thành phần backend dùng ngôn ngữ lập trình Node.js và thành phần frontend dùng ngôn ngữ lập trình Svelte. Budibase được xác định là phù hợp để xây dựng các hệ thống thông tin của doanh nghiệp nhỏ và vừa.

Appsmith

Appsmith là nền tảng LCDP có cách tiếp cận tương tự như Budibase để hỗ trợ phát triển các ứng dụng nội bộ trong doanh nghiệp nhỏ và vừa. Tuy nhiên Appsmith không hỗ trợ công cụ thiết kế CSDL dùng riêng mà chỉ dùng các kết nối dữ liệu ngoài bao gồm các CSDL SQL, NoSQL, API và SMTP. Ngoài ra, công cụ hỗ trợ thiết kế quy trình nghiệp vụ (workflow) cho ứng dụng. Tuy nhiên công cụ này chưa được hỗ trợ đối với phiên bản sử dụng cho cộng đồng. Chính vì vậy việc khai thác phần mềm nguồn mở Appsmith bị gặp nhiều hạn chế.

Tooljet

Cùng cách tiếp cận tương tự như Budibase và Appsmith, Tooljet là nền tảng mã nguồn mở LCDP hỗ trợ phát triển ứng dụng hướng thành phần. Tooljet cũng cung cấp ba bộ công cụ hỗ trợ bao gồm tạo dựng CSDL, thiết kế giao diện ứng dụng, và mô hình hóa luồng công việc (workflow). Điểm khác biệt của Tooljet nằm ở khả năng hỗ trợ trực quan hóa quy trình xử lý dưới dạng đồ họa biểu diễn dạng đồ thị. Tuy nhiên công cụ này (bản beta) hiện chưa được cung cấp cho cộng đồng dưới hình thức nguồn mở.

So sánh với các nền tảng nguồn mở khác, số lượng các loại kết nối dữ liệu mà Tooljet hỗ trợ rất đa dạng bao gồm đủ các kiểu CSDL (SQL, NoSQL, BigData), các dịch vụ ứng dụng (SaaS), các dịch vụ lưu trữ đám mây (S3, Azure, Google), các kết nối API (GraphQL, Restful, gRPC). Người dùng cũng có thể tổ chức lưu trữ dữ liệu của ứng dụng trên CSDL được thiết kế bằng chính công cụ Tooljet. Các thành phần đồ họa của ứng dụng có thể được mở rộng theo nhu cầu bằng cách sử dụng ngôn ngữ lập trình React.

NocoBase

NocoBase là một nền tảng LCDP mã nguồn mở mới được xây dựng gần đây nhưng có rất nhiều tiềm năng phát triển. NocoBase sở hữu một kiến trúc hoàn toàn mở dựa trên phương pháp phát triển ứng dụng hướng mô hình (MDD). Nền tảng cung cấp ba bộ công cụ hỗ trợ người dùng bao gồm: thiết kế mô hình hóa dữ liệu (data model), thiết kế quy trình nghiệp vụ (workflow), và thiết kế giao diện người dùng (UI). Ứng dụng được phát triển rất linh hoạt nhờ sự phân tách hoàn toàn giữa thành phần backend (dịch vụ dữ liệu và logic nghiệp vụ) và frontend (giao diện tương tác người dùng).

Mọi công cụ trên nền tảng NocoBase đều được phát triển theo kiến trúc sử dụng plugins với các ngôn ngữ lập trình hiện đại gồm Node.js, React, Koa. Do đó nó cho phép nhà phát triển mở rộng chức năng của nền tảng bằng cách tạo mới các plugin được tùy chỉnh theo nhu cầu. Phiên bản nguồn mở dùng cho cộng đồng cung cấp sẵn các plugins cơ bản, thiết yếu trong phát triển ứng dụng. Các plugins có tính năng nâng cao được phân phối dưới hình thức bán bản quyền thương mại. NocoBase là nền tảng có khả năng tùy biến, mở rộng cao, phù hợp với các dự án lớn và phức tạp.

Joget DX

Joget DX là một nền tảng LCDP mã nguồn mở hỗ trợ phát triển ứng dụng với giao diện được lập trình theo phương pháp hướng thành phần (CDD). Nó cung cấp bốn công cụ dùng để thiết kế ứng dụng theo mô hình bao gồm: công cụ thiết kế giao diện biểu mẫu (Form Builder), công cụ thiết kế giao diện danh sách (List Builder), công cụ thiết kế giao diện màn hình ứng dụng (UI Builder) và công cụ thiết kế quy trình xử lý nghiệp vụ (Process Builder). Tất cả các công cụ đều sử dụng màn hình hiển thị trực quan để người dùng thực hiện các thao tác cấu hình thay vì phải viết mã. Người dùng có thể kéo thả các thành phần để tạo dựng và tùy biến các giao diện dạng biểu mẫu của ứng dụng. Ngoài các thành phần có sẵn trong nền tảng, nhà phát triển có thể lập trình bổ sung các thành phần mới dưới dạng các plugins được viết mã bằng ngôn ngữ Java.

Joget DX có mục đích sử dụng như là một nền tảng phục vụ chuyển đổi số. Đi cùng với nền tảng LCDP này, có một chợ chia sẻ (marketplace) các ứng dụng và plugins đã được phát triển. Hiện đang có 70+ ứng dụng và 110+ plugins được công bố trên chợ. Chúng được xây dựng bởi chính Joget, các đối tác và cộng đồng nguồn mở. Các tổ chức, doanh nghiệp có thể rút ngắn thời gian, chi phí để thực hiện chuyển đổi số khi có thể khai thác, sử dụng lại các ứng dụng và plugins được chia sẻ trên chợ của nền tảng phát triển.

Convertigo

Convertigo là một nền tảng LCDP mã nguồn mở có bốn thành phần kiến trúc là Server, Studio, SDKs và Forms. Convertigo Server là thành phần backend dạng mBaaS dùng để xử lý kết nối, thực thi các vi dịch vụ và đồng bộ hóa dữ liệu offline cho thiết bị di động. Convertigo Studio là một IDE dựa trên Eclipse, được sử dụng để tạo lập quy trình cho các vi dịch vụ chạy trên Server. Ngoài ra nó còn cung cấp tiện ích hỗ trợ thiết kế giao diện ứng dụng di động ở chế độ ít mã. Convertigo SDKs được cung cấp cho các nhà phát triển để tự do phát triển ứng dụng frontend bằng ngôn ngữ lập trình ưa thích như Angular, React Native và Vue.js. Convertigo Forms là công cụ xây dựng giao diện ứng dụng không cần mã (no-code) dành cho cá nhà phát triển không chuyên.

Các ưu điểm của Convertigo là khả năng hỗ trợ phát triển ứng dụng đa nền tảng. Nền tảng này cung cấp dịch vụ kết nối với nhiều loại nguồn dữ liệu khác nhau bao gồm CSDL SQL,

NoSQL, Rest API, SOAP và dịch vụ SAP. Các ứng dụng di động được xây dựng bằng Convertigo có thể hoạt động offline, giúp tăng trải nghiệm người dùng.

Axelor

Axelor là một nền tảng LCDP mã nguồn mở hỗ trợ phát triển ứng dụng trong doanh nghiệp một cách toàn diện. Thành phần lõi của Axelor là một nền tảng phát triển ứng dụng hướng mô hình (MDD) được dùng cho các nhà phát triển chuyên sâu có hiểu biết về ngôn ngữ lập trình Java. Để đáp ứng nhu cầu sử dụng ít mã hoặc không mã, Axelor cung cấp một bộ công cụ Studio để hỗ trợ người dùng không chuyên có thể thiết kế trực quan giao diện ứng dụng (UI) và quy trình nghiệp vụ (BPM) bằng hình thức kéo thả. Bộ công cụ trực quan này sẽ tự động tạo mã được dùng bởi thành phần lõi nền tảng phát triển Axelor.

Axelor đồng thời cung cấp một giải pháp ERP toàn diện mã nguồn mở cho doanh nghiệp trong một bộ các ứng dụng được thiết lập sẵn (Alexor Suite). Các ứng dụng phủ đầy đủ các lĩnh vực như hành chính văn phòng, tài chính, kinh doanh, chăm sóc khách hàng, nhân sự, quản lý dự án, sản xuất, chuỗi cung ứng. Ngoài ra, khách hàng cũng có thể lựa chọn sử dụng mua bản quyền thương mại để có thể sử dụng thêm các công cụ hỗ trợ khác như thông minh nghiệp vụ (BI), kết nối tích hợp ứng dụng với bên thứ ba. Tất cả đều được xây dựng hướng tới đối tượng người dùng trong doanh nghiệp, không có kiến thức chuyên sâu để viết mã lập trình.

3.2 Các nền tảng LCDP chuyên dùng

Hiện vẫn còn ít nền tảng LCDP đa dụng mã nguồn mở được cung cấp trên thị trường như đã trình bày ở trên. Ngoài ra các tính năng được cung cấp bởi các nền tảng mã nguồn mở cũng chưa đủ mức toàn diện như các nền tảng thương mại được đánh giá công nhận bởi Gartner trong Magic Quadrant. Chính vì vậy mà phạm vi ứng dụng của các nền tảng mã nguồn mở này đang vẫn còn bị hạn chế. Tuy nhiên, phần mềm nguồn mở lại có sự phát triển mạnh trong việc cung cấp các nền tảng LCDP chuyên dùng thường đi theo các mục tiêu ứng dụng trong lĩnh vực hẹp. Dưới đây là một sát khảo sát các nền tảng mã nguồn mở theo từng nhóm tính năng.

Nền tảng quản lý nội dung tách rời với trình bày (Headless CMS)

Headless CMS là nền tảng quản lý nội dung tập trung vào lưu trữ và quản lý nội dung, chứ không phải cách hiển thị nội dung đó. Khác với các hệ thống quản lý nội dung truyền thống (CMS), nền tảng không đi kèm với công cụ thiết kế giao diện frontend cố định dùng để xuất bản các nội dung. Từ đó nội dung có thể được xuất bản trên nhiều nền tảng khác nhau (Web, Mobile, IoT, v.v.). Nhà phát triển có thể tự do lựa chọn các công nghệ lập trình ưa thích và phù hợp cho ứng dụng frontend.

Strapi là một nền tảng Headless CMS mã nguồn mở, được xây dựng trên Node.js. Nó cho người dùng quản lý nội dung một cách trực quan thông qua một giao diện quản trị, nhưng không giới hạn bạn vào một giao diện frontend cụ thể. Strapi cung cấp các API REST và GraphQL đầy đủ tính năng, giúp nhà phát triển tích hợp với các ứng dụng frontend khác. Đồng thời người dùng có thể mở rộng chức năng của ứng dụng một cách dễ dàng thông qua các plugins được chia sẻ trên chợ ứng dụng.

Directus là một nền tảng nguồn mở khác hoạt động như một Headless CMS. Nó cung cấp một giao diện người dùng thân thiện để tạo các bảng, trường, quan hệ và thực hiện các

thao tác quản lý dữ liệu khác. Người dùng đồng thời có thể tùy chỉnh giao diện, tạo các view và báo cáo theo ý muốn. Directus có thể thay thế các bảng điều khiển, giúp tiết kiệm thời gian và công sức của quản trị viên đối với các CSDL dạng SQL (MySQL, PostgreSQL, SQLite,...).

Nền tảng cung cấp dịch vụ Backend (BaaS)

Mục đích xây dựng của loại nền tảng này là các dịch vụ và công cụ cần thiết để xây dựng và vận hành phần backend của một ứng dụng. Từ đó nhà phát triển có thể sử dụng tới rất ít mã lập trình để tạo ra các API cho ứng dụng. Nó là xu thế phát triển mới đi theo theo mô hình kiến trúc microservice cho phép tạo ra nhiều loại hình frontend khác nhau (Web, Mobile, IoT) cho cùng một ứng dụng. Các nền tảng BaaS dùng ít mã thường có cách tiếp cận theo phương pháp phát triển ứng dụng hướng mô hình (MDD).

Supabase và Parse là hai nền tảng mã nguồn mở được xây dựng như là một thay thế cho Firebase. Chúng đều cung cấp công cụ tạo dựng CSDL và sinh tự động API được dùng cho ứng dụng. Đi kèm với đó là bộ thư viện SDK được sinh tự động để hỗ trợ người dùng phát triển nhiều loại frontend bằng các ngôn ngữ khác nhau (Javascript, Flutter, Swift, Kotlin, Python, C#,...). Parse có ưu điểm là hỗ trợ mô hình hóa hướng đối tượng để có thể lưu trữ dữ liệu trong cả hai loại CSDL là Postgres và MongoDB. Trong khi đó, Supabase tập trung vào việc kiểm soát hoàn toàn một loại CSDL chuyên dùng là Postgres với thiết kế bằng mô hình quan hệ. Ngoài ra, Supabase còn hỗ trợ công cụ tạo hàm dịch vụ phi máy chủ (serverless) để hoạt động theo mô hình đám mây FaaS (Function as a Service).

Rowy là một nền tảng BaaS tập trung vào sự đơn giản và dễ sử dụng. Nó hỗ trợ người dùng xây dựng ứng dụng và cơ sở dữ liệu thông qua giao diện trực quan. Rowy phù hợp cho các dự án cần phát triển nhanh chóng, không yêu cầu kỹ năng lập trình sâu và muốn một giao diện thiết kế trực quan.

Amplication là một nền tảng BaaS được thiết kế theo cách tiếp cận phát triển hướng mô hình (MDD) để tạo ra các API, mô hình dữ liệu, và logic nghiệp vụ của ứng dụng. Dựa trên mô hình dữ liệu đầu vào, nền tảng hỗ trợ tự động tạo mã API REST, GraphQL, dịch vụ xác thực, lưu vết (logging) và giao diện quản trị dùng cho backend của ứng dụng. Người dùng cũng có thể sử dụng lại lược đồ từ CSDL hiện có để bắt đầu phát triển nhanh hơn (PostgreSQL, MySQL, MongoDB, MS SQL Server,…). Ngoài ra, Amplication cũng hỗ trợ phát triển ứng dụng hướng sự kiện (EDD) với mô hình kiến trúc microservice. Amplication có tính mở cao, người dùng có thể tùy chỉnh, mở rộng các chức năng bằng cách sử dụng các plugins và thư viện của bên thứ ba. Đặc biệt hơn, Amplication được tích hợp sẵn với OpenAI để cung cấp tính năng hướng dẫn trợ giúp và tự động sinh mã dựa trên ý tưởng của người dùng.

Nền tảng mô hình hóa quy trình nghiệp vụ (BPM)

Mục đích xây dựng của loại nền tảng này là cho phép các tổ chức, doanh nghiệp mô hình hóa, tự động hóa và tối ưu hóa các quy trình nội bộ của đơn vị mà không cần phải viết quá nhiều mã. Với giao diện trực quan và các công cụ kéo thả, người dùng có thể dễ dàng thiết kế, quản lý và theo dõi các quy trình phức tạp, từ công việc của các chuyên viên đến quyết định phê duyệt của lãnh đạo. Nền tảng sử dụng các biểu đồ và sơ đồ mô tả các bước trong quy trình để gắn các tác vụ cho người dùng nhằm theo dõi tiến độ hoàn thành.

Flowable là một nền tảng BPM mã nguồn mở được thiết kế để mô hình hóa, thực thi và quản lý quy trình dựa trên các ngôn ngữ mô hình hóa tiêu chuẩn mở gồm BPMN, CMMN và DMN. Chúng được thực thi với một engine mã nguồn mở dùng miễn phí trong phiên bản cộng

đồng giống như của Camunda, Bonita hay Activiti. Các phần mềm này đều có nguồn gốc phát triển từ thư viện jBPM sử dụng ngôn ngữ Java. Chúng cho phép mô hình hóa, thực thi, giám sát và tối ưu hóa các quy trình nghiệp vụ bằng các công cụ trực quan như trình thiết kế quy trình, trình thiết kế biểu mẫu, trình thiết kế giao diện người dùng và trình thiết kế kịch bản.

Nền tảng quản lý tự động hóa quy trình và tác vụ (Automation)

Mục đích xây dựng của loại nền tảng này là hỗ trợ tự động hóa các tác vụ lặp đi lặp lại, tiết kiệm thời gian và tăng năng suất làm việc. Tại đây, phương pháp phát triển ứng dụng hướng sự kiện (EDD) chủ yếu được áp dụng để thực hiện kết nối tự động nhiều dịch vụ khác nhau trong cùng một quy trình làm việc thống nhất.

N8N là một nền tảng tự động hóa quy trình hoạt động dựa trên khái niệm luồng công việc (workflow). Một workflow là một chuỗi các hành động (nodes) được kết nối với nhau để thực hiện một nhiệm vụ cụ thể. Mỗi node đại diện cho một tác vụ riêng biệt, như gửi email, tải lên file, trích xuất dữ liệu từ một trang web, v.v. Bằng cách kéo và thả các node vào một màn hình và kết nối chúng lại với nhau thành một đồ thị, người dùng có thể tạo ra một workflow phức tạp để tự động hóa quy trình xử lý theo yêu cầu. N8N là nền tảng mã nguồn mở cung cấp sẵn các bộ kết nối (plugins) tới hầu hết các loại dịch vụ được sử dụng phổ biến hiện nay thông qua API. Người dùng có thể viết mã tùy chỉnh cho các nút tác vụ bằng ngôn ngữ Javascript.

PipeDream là phần mềm nguồn mở có các tính năng tương tự như N8N. PipeDream được thiết kế để phù hợp hơn với các lập trình viên bởi tính linh hoạt cao với khả năng kiểm soát hoàn toàn đối với các workflow được tạo ra. Nó đáp ứng đối với các trường hợp cần giải pháp linh hoạt và có thể tùy biến cao.

Huginn là một nền tảng mã nguồn mở cho phép người dùng tạo ra các tác tử (agent) để tự động hóa các tác vụ trực tuyến. Những agent này có thể đọc thông tin từ web, theo dõi các sự kiện và thực hiện các hành động thay mặt cho người dùng. Các agent được thiết lập sự tương tác với nhau thông qua các sự kiện (event). Một agent có thể tạo ra một event, và các agent khác có thể lắng nghe và phản hồi lại event đó. Từ đó, các agent có thể hoạt liên kết với nhau để đáp ứng nhu cầu tự động hóa công việc cho cá nhân hoặc tổ chức.

StackStorm là một nền tảng mã nguồn mở được xây dựng chuyên biệt cho tự động hóa các quy trình quản lý vận hành thuộc lĩnh vực IT (hoạt động DevOps). Khác biệt chính của StackStorm là tập trung vào thực hiện các tác vụ hoặc chạy các quy trình dựa trên sự kiện. Người dùng cần kiến thức lập trình về Python để tận dụng tối đa khả năng tùy chỉnh, mở rộng của nền tảng.

Node-RED là một nền tảng mã nguồn mở khác được xây dựng chuyên biệt cho tự động hóa quy trình trong lĩnh vực IoT. Cơ chế hoạt động của Node-Red tương tự như N8N. Tuy nhiên nó có sự khác biệt là hỗ trợ 4000+ bộ kết nối nguồn dữ liệu và giao thức sử dụng trong IoT như Modbus, OPC-UA, Siemens S7 và MQTT. Node-RED rất linh hoạt và có khả năng mở rộng cao, phù hợp với các nhà phát triển chuyên sâu.

Nền tảng quản lý dữ liệu dạng bảng tính thông minh

Mục đích xây dựng của loại nền tảng này là quản lý lưu trữ và truy xuất dữ liệu trong các CSDL SQL thông qua các giao diện trực quan như một bảng tính thông minh. Các tính năng cơ bản của nền tảng bao gồm: tạo mới các bảng dữ liệu với các trường tùy chỉnh một cách dễ dàng; nhập, xuất, chỉnh sửa, xem, tìm kiếm, lọc dữ liệu trực tiếp trên giao diện dạng bảng tính;

trực quan hóa dữ liệu bảng tính bằng các biểu đồ khác nhau (biểu đồ cột, biểu đồ đường, biểu đồ tròn, v.v.); tùy biến giao diện hiển thị dữ liệu theo dạng mong muốn.

NocoDB, Baserow là hai nền tảng mã nguồn mở được xây dựng để thay thế cho Airtable. Chúng cho phép kết nối lưu trữ với bất kỳ cơ sở dữ liệu SQL nào (vd., MySQL, PostgreSQL, SQL Server, SQLite, MariaDB,…). Người dùng có thể sử dụng giao diện trực quan để quản lý và thao tác dữ liệu một cách hiệu quả, mà không cần phải viết nhiều câu lệnh SQL phức tạp. Baserow có giao diện hiện đại, dễ thao tác, trong khi NocoDB có giao diện đơn giản, dễ hiểu. Ngoài ra Baserow có cung cấp thêm công cụ xây dựng ứng dụng (Application Builder) cho người dùng.

4. Chuẩn bị cho cuộc thi PMNM - OLP 2024

Cuộc thi phần mềm nguồn mở do Hội Tin học Việt Nam tổ chức hàng năm cho sinh viên của các trường đại học, cao đẳng trên cả nước tham gia. Thể lệ của cuộc thi năm 2024 đã được công bố trên trang thông tin của Ban tổ chức. Hình thức của cuộc thi là các đội tuyển tham gia làm một dự án phát triển phần mềm nguồn mở theo yêu cầu của đề thi do BTC công bố.

Chủ đề của cuộc thi năm 2024 là “Nền tảng phát triển ứng dụng dùng ít mã nguồn - LCDP". Mục đích của chủ đề là giúp sinh viên nắm bắt được xu thế sử dụng nền tảng trong phát triển ứng dụng phục vụ chuyển đổi số các doanh nghiệp, tổ chức. Qua đó sinh viên làm chủ được công nghệ và các kỹ năng cần thiết để thích ứng với các yêu cầu phải rút ngắn thời gian, chi phí đầu tư, duy trì các hệ thống ứng dụng CNTT trong tương lai. Phần mềm nguồn mở luôn là một lựa chọn tốt để bắt đầu học hỏi, tiếp cận sử dụng cho các nhu cầu phát triển cả tư duy và kỹ năng.

Các trường tham gia cuộc thi cần lựa chọn sinh viên để tham gia hoạt động huấn luyện đội tuyển với các kĩ năng, kiến thức cần thiết sau đây:

- Hiểu biết về dự án xây dựng PMNM: Nắm vững và có kĩ năng sử dụng các công cụ hỗ trợ trong xây dựng một dự án PMNM đáp ứng được các tiêu chí cần thiết theo đúng thông lệ quốc tế.
- Nắm vững các phương pháp phát triển ứng dụng dùng ít mã: Tư duy lập trình dùng ít mã được đã tổng kết bằng bốn phương pháp chính như đã trình bày ở trên. Các phương pháp này không phụ thuộc vào bất kì ngôn ngữ lập trình hoặc một nền tảng phát triển cụ thể.
- Làm quen, thực hành sử dụng các nền tảng mã nguồn mở: Đề thi PMNM của năm sẽ yêu cầu phát triển ứng dụng dựa trên một nền tảng phát triển dùng ít mã do các đội tham gia tự lựa chọn. Nền tảng sử dụng bắt buộc phải là PMNM. Đồng thời nền tảng được lựa chọn cũng phải có sự phù hợp cho phép đóng gói ứng dụng tạo ra dưới dạng mã để lưu trữ, chia sẻ trên một kho mã nguồn công khai.
- Sáng tạo trong giải quyết vấn đề: Khả năng tùy biến, mở rộng theo yêu cầu người dùng đặc biệt được quan tâm trong các nền tảng LCDP. Các đội thi được khuyến khích việc tìm hiểu, sáng tạo để đóng góp thêm các tính năng cho một nền tảng LCDP thông qua các plugins mới được tạo ra. Đặc biệt trong quá trình giải quyết vấn đề, các đội thi có thể khai thác thêm công cụ AI để trợ giúp sinh mã nguồn cho nền tảng. Tất cả các sáng tạo có thể được nghiên cứu, thực hành từ trước khi đề thi chính thức được công bố.