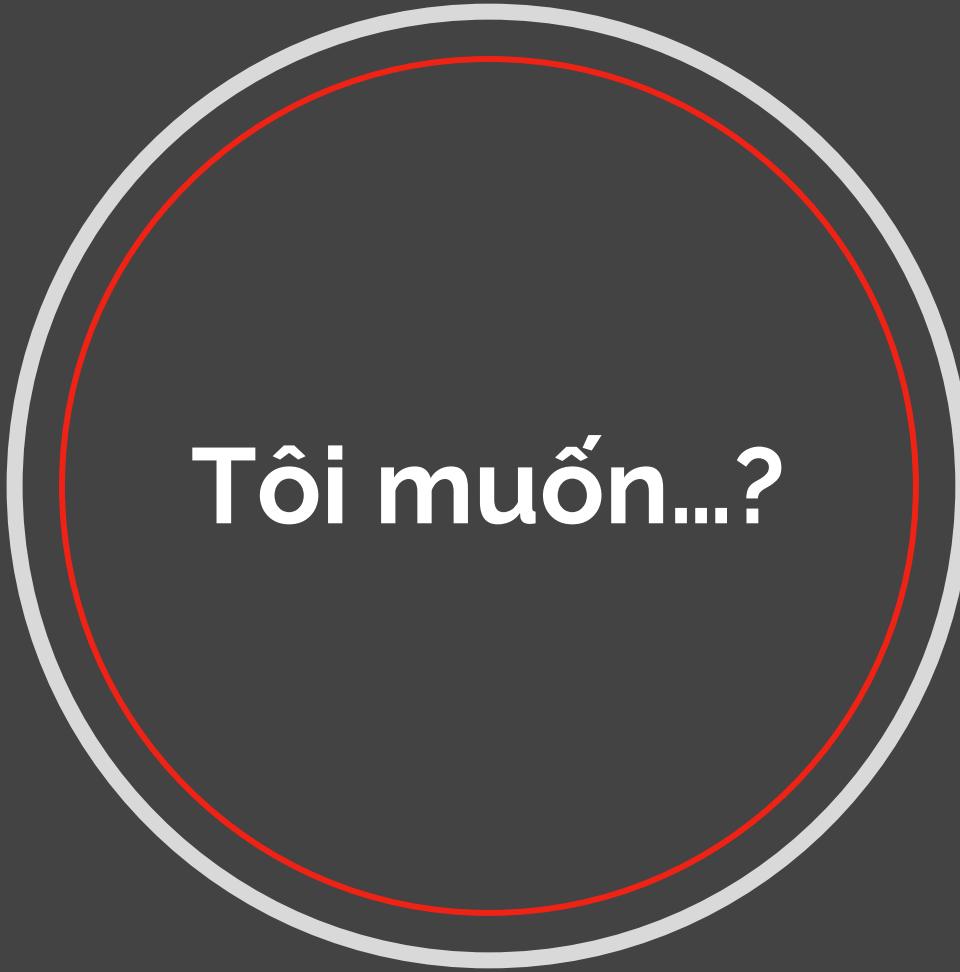




# Xây dựng codebase Universal

Hay là câu chuyện về làm sao để nâng cao đời sống tinh thần của DEV





Tôi muốn...?

# MỤC TIÊU

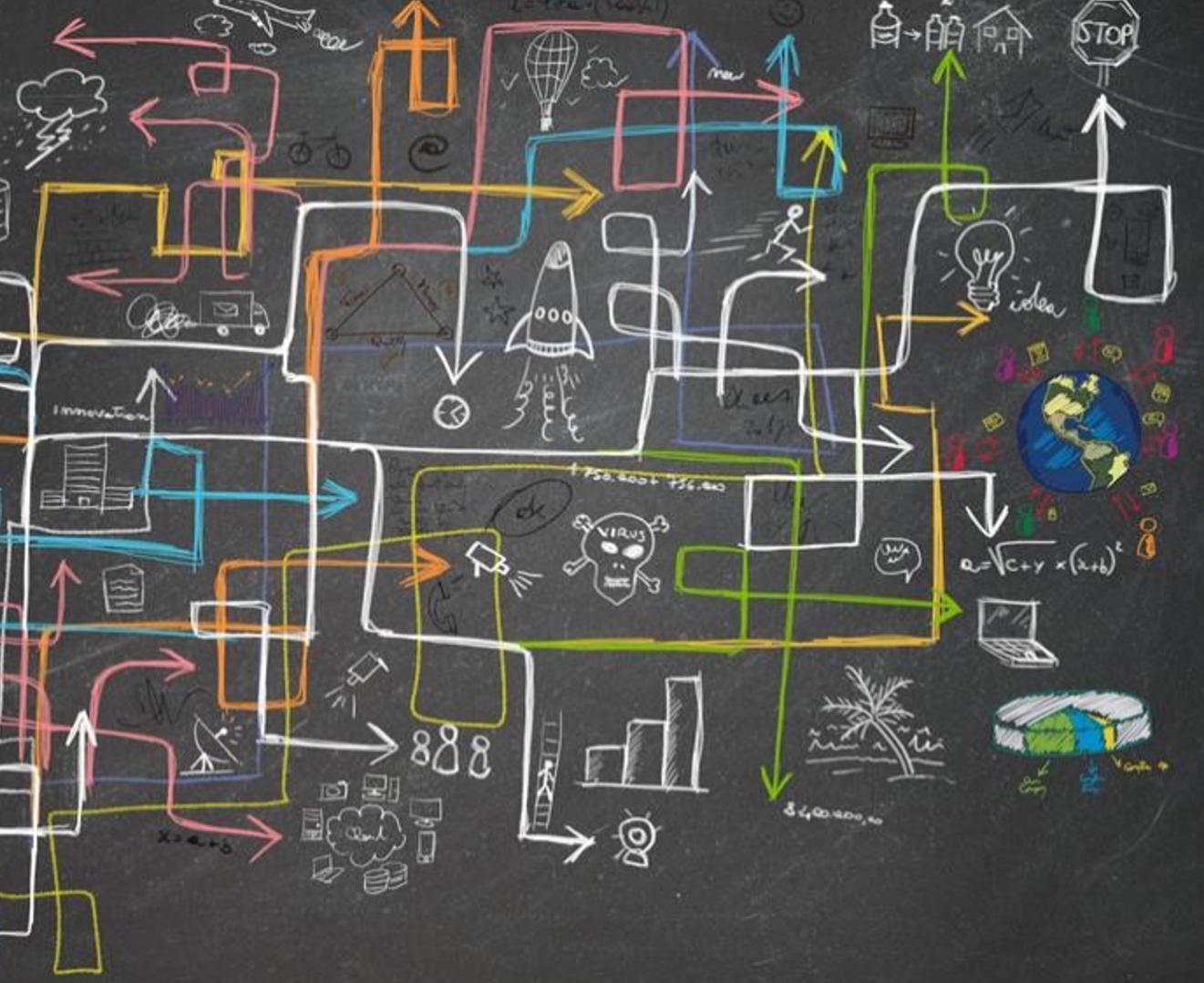
- Chia sẻ về codebase được xây dựng và làm rõ tư duy, cách làm, cách phát triển và vận hành
- Chia sẻ về Roadmap cũng như các Architect xưa đến nay
- Thảo luận về cách thức đóng gói phần mềm, xây dựng lộ trình phát triển kỹ năng cho Dev và mở rộng mạng lưới mối quan hệ cho Dev.
- Nâng cao Performance, bảo mật cho hệ thống
- Giới thiệu 1 số khái niệm cơ bản về Test, SEO khi làm Web

# Tổng quan buổi

1

## Thông tin bối cảnh dự án, tình hình source code

## 1 vài workshop xung quanh công việc của DEV





# Speaker: HungNA - PHP Advisor

- ~12 năm kinh nghiệm lập trình PHP, 8 năm kinh nghiệm Lead-Manager, 4 năm kinh nghiệm PM/PO
- Trong 2 năm build team dev từ zero lên ~50 member (chính thức) với nhiều ngôn ngữ: PHP, NodeJs, React, Python
- Phụ trách mảng Payment Gateway: Telco, Ví điện tử, nạp thẻ với khoảng 40 triệu khách hàng
- Xây dựng và phát triển mảng News CMS của công ty triển khai tới ~100 trang báo / cổng thông tin điện tử
- Triển khai hệ thống Voice-to-text kết nối giải pháp cho Viettel với Nuance, phục vụ tính năng gọi nhỡ, tin nhắn thoại cho tập khách hàng ~70 triệu của viettel
- Tham gia xây dựng và phát triển giải pháp Sigma OTT, Sigma DRM, Sigma Securities
- Thiết kế kiến trúc hệ thống, lead team dev xây dựng coolmate.me - Platform bán hàng cho nam giới phục vụ lượng đơn hàng cao điểm lên gần 30k đơn hàng / ngày, bị DDOS thường xuyên :))
- Tham gia vào việc xây dựng quy chuẩn code (Application Standard) và Codebase của Beetsoft
- Trưởng phòng công nghệ và quản lý dự án tại Beetsoft

# PHP 2023

PHP là ngôn ngữ lập trình thủ tục và hướng đối tượng có tuổi đời lớn (từ 1995)

Hiện tại đã phát triển tới phiên bản PHP 8

Điều gì đầu tiên xuất hiện trong đầu bạn khi nhắc tới PHP

# 1 vài điểm mới trong PHP

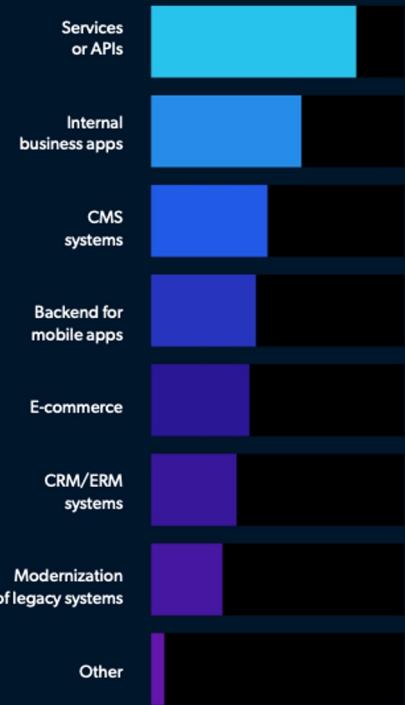
## Ngày xưa

- Dùng làm web, blog... tiêu biểu là WordPress, Joomla
- Chỉ làm những sản phẩm nhỏ
- PHP lỏng lẻo, thiếu bảo mật và hiệu năng thấp
- PHP chỉ hoạt động trên nền web
- PHP là ngôn ngữ đồng bộ, không realtime

## Nay

- Theo báo cáo mới nhất từ tổ chức Zend, PHP nay đã phong phú hơn nhiều: Services API, CRM/ERM system, CMS system, Backend for Mobile App, Ecommerce
- Rất nhiều dự án lớn sử dụng PHP. Rất nhiều framework mới được ra đời
- 70% thị phần website thuộc về PHP, công lớn thuộc về WordPress
- Các phiên bản sau PHP càng nhiều đặc điểm thú vị hơn: định kiểu, bất đồng bộ, realtime.
- Từ phiên bản PHP 8, tốc độ của PHP đã lên tầm mới với bộ biên dịch mới, ở 1 số bài test, PHP 8 cho tốc độ tốt hơn cả NodeJS, Ruby hay Java, kém Golang 1 chút
- Rất nhiều platform Low-code/No-code như wix, weebly ra đời khiến cho mảng đất website đơn giản có nhiều sự lựa chọn hơn với việc chỉ cần kéo thả cũng đã có ngay 1 website xịn sò
- Phối hợp tốt hơn với nhiều frontend framework như vuejs, reactjs, angularjs

**Q: What Are You Using PHP For?  
(Select All That Apply.)**

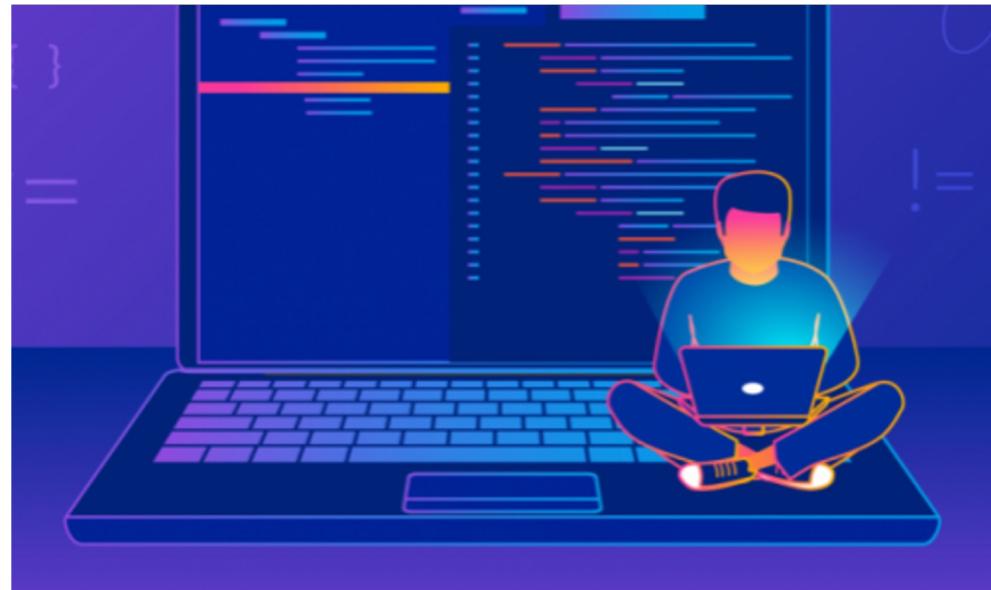


# Overview

---

Chung quy lại thì:

- Các website nhỏ dạng giới thiệu công ty, trường học, sản phẩm hết đất sống với các nền tảng no-code, low-code. Duy nhất WordPress sống được với mảng này, tuy nhiên nó là 1 phạm trù riêng biệt
- Nhiều framework hơn, khẩu vị khách hàng mỗi bên 1 kiểu, dev cần phải học nhiều hơn xưa
- Làm nhiều thể loại hơn yêu cầu kiến thức phong phú hơn: rest, services, erp, crm
- Vừa phải học nhiều hơn vừa phải cải tiến liên tục



---

PHP @ 2023



# Điểm danh

## Dự án

Có 1 số dự án sử dụng PHP vẫn đang được các bạn maintain như Nakajimax, Sixcloud, Kenkamichi ...

1

2

## Codebase

Hầu như chưa có, tài liệu rời rạc. Chủ yếu sử dụng Laravel

3

4

## Tìm điểm ra

Xây dựng quy chuẩn, codebase, tài liệu

...n

## Thực trạng các job

Ít job, từ lúc vào đến h chủ yếu maintain và làm codebase mặc dù estimate khá nhiều

## Nhân sự

Tương quan năng lực rời rạc chưa đồng nhất, tốn nhiều thời gian để tiếp cận cái mới

# Think Tank



## Nội bộ

Thống nhất quy cách làm việc, tiêu chuẩn nội bộ và liên tục xây dựng + làm đầy được know-how + thư viện lập trình nội bộ -> tăng hiệu suất, hiệu quả + chất lượng cho toàn đội phát triển.

## Thị trường

Xây dựng được lợi thế cạnh tranh cực lớn với các cty khác cùng ngành. Với BPDS ta tự tin làm hài lòng mọi k/h với tiêu chuẩn cao nhất và chất lượng tốt nhất thị trường.



## Codebase, Document & Training

Phân tích, triển khai bộ codebase và tiêu chuẩn cho các team

# Cách truyền thống

## Xây dựng codebase

Team vẽ mô hình, áp dụng clean code, design pattern các kiểu, bao gồm backend, frontend. Hoàn thành 1 base project, lưu trên git repository như github, gitlab, bitbucket. Đặt tên là **codebase**

1

2

3

4

..n

## Sử dụng codebase cho dự án mới (lần đầu)

Clone repository **codebase** về, xây dựng các feature mới trong repository. Ở bước này 1 số tính năng trong codebase giữ nguyên. Deploy, hoàn thành, đặt tên là **project01**

## Tiếp tục phát triển...

Lặp lại bước 3. Mỗi dự án modify feature 1 chút cho phù hợp nhu cầu. Cơ bản, code ở bước này chỉ còn liên quan mảng với **codebase** ban đầu

## Tiếp tục phát triển dự án tiếp theo

Copy code **project01**, chỉnh sửa lại 1 số tính năng trong các module base cho phù hợp. Add thêm 1 số tính năng mới. Đặt tên là **project02**

## Kết thúc vòng đời codebase

Dự án thứ n, code nát, các dự án không maintain nữa, lead và dev gánh team mãi không chịu được đã nghỉ. Tuyển team mới vào xây codebase từ đầu

---

# Ưu / nhược điểm của cách làm truyền thống

## Ưu điểm:

- Ở bước codebase, độc senior làm việc nén tài liệu, test siêu đầy đủ, code siêu đẹp
- Triển khai dự án đầu tiên: rất nhanh

## Nhược điểm:

- Cửa sổ vỡ có thể xuất phát ở bất kì đâu
- Codebase là out-date, nếu muốn codebase được update thì nguồn lực để đánh giá, copy là rất lớn
- Giả sử, module gốc trong codebase có bug, cần fix và có khoảng 10 dự án đã sử dụng đoạn codebase đó (vài dòng thôi) thì chi phí update cũng tốn thời gian lớn
- Khả năng tận dụng nguồn lực dạng scrum bị hạn chế: dev dạng intern và fresher thường sẽ không có cơ hội contribute vào codebase
- Khi nhiều người cùng làm chung một dự án, đôi khi mỗi người sử dụng một phiên bản riêng, xung đột lẫn nhau. Code chạy được ở máy này, không chạy được ở máy khác.



# Cách tôi đang triển khai

## Xây dựng codebase

Team vẽ mô hình, các phân hệ cần thiết, các tính năng cần thiết trong codebase. Đóng gói và publish/private thành package. Link các package lại thành 1 base project. Đặt tên **codebase**

1

2

3

4

..n

## Đóng gói feature

Các feature được đóng gói thành các package riêng, ví dụ: bs-user, bs-roles, bs-permission, bs-login, bs-social-sso, bs-jwt... Các feature được chia nhỏ thành các repo riêng, vận hành và có mô hình tài liệu, document, maintainer riêng

## Phát triển dự án

Fork dự án codebase, xem xét các feature của dự án xem các package đã có chưa. Nếu đã có sẽ required thông qua package manager. Nếu có nghiệp vụ mới có tiềm năng thì sẽ tạo thêm package mới. Các feature thực tế được tổ chức ở các package thay vì code repo code

## Refactoring, document, test

Mỗi khi có thời gian rảnh, cả team sẽ ngồi với nhau nhìn lại xem các package có cần nâng cấp gì, bổ sung feature nào mới hay không, có nợ kỹ thuật nào cần trả hay không để bổ sung

## Tiếp tục phát triển...

Song hành với quá trình phát triển dự án, liên tục nâng cấp các package theo chuẩn semantic versioning, đồng thời đóng gói các package mới ở các dự án mới, đưa thành codebase chờ dự án tiếp theo có feature tương tự cần dùng tới

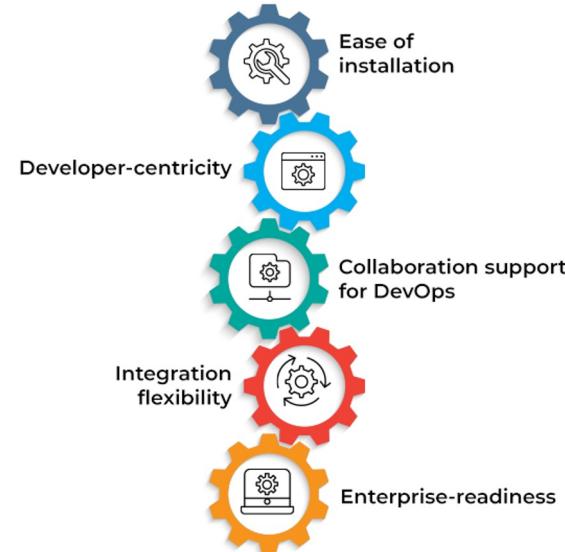
# Ưu / nhược điểm của cách làm mới

## Ưu điểm:

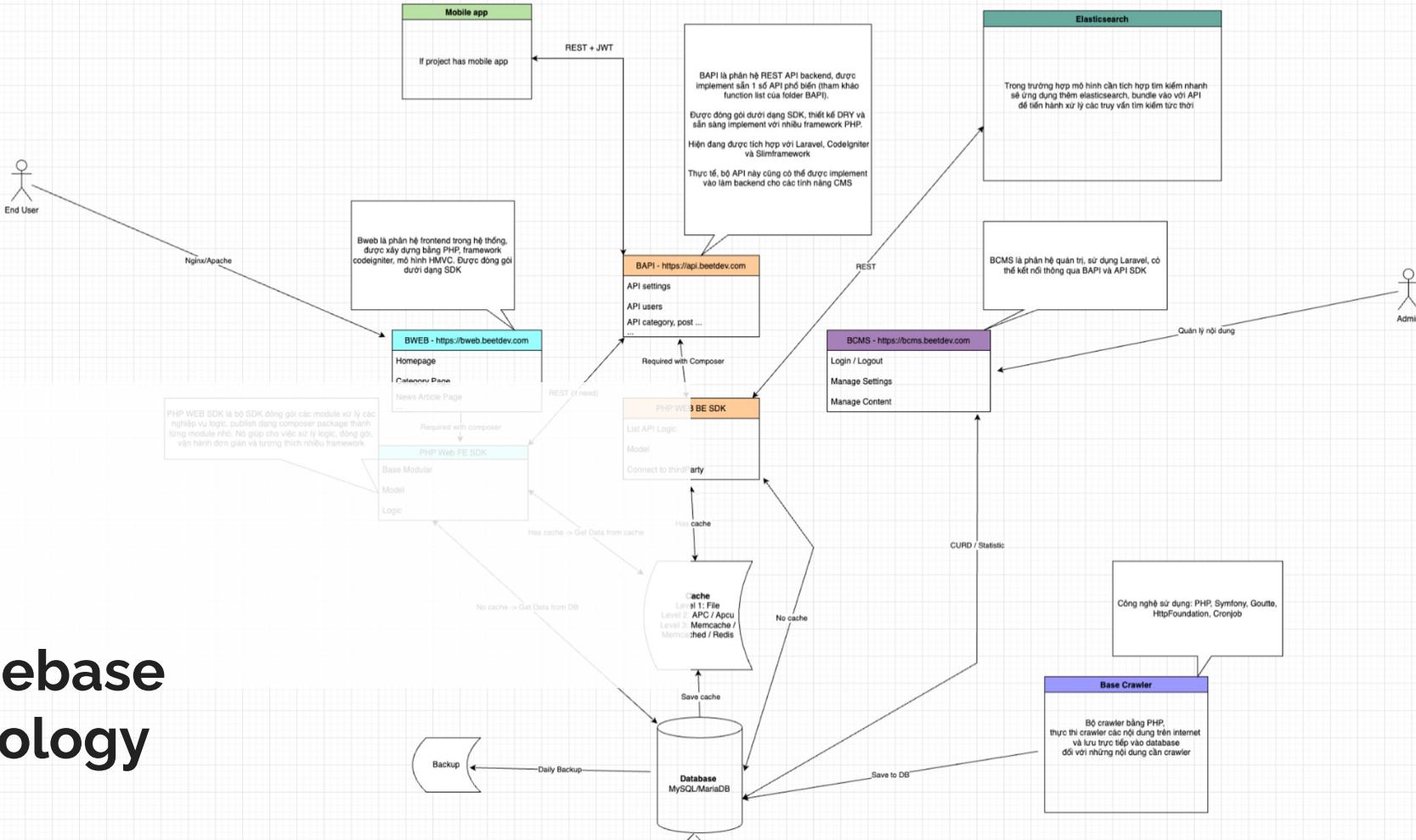
- Codebase được đóng gói, phát triển liên tục. Có thể sử dụng versioning để duy trì nhiều phiên bản của 1 package. Codebase vì thấy luôn là bản code đầy đủ, có khả năng bao quát nhiều tính năng
- Bất cứ ai trong team cũng có cơ hội join làm codebase, việc phụ trách 1 package giúp dev có khả năng phát triển kỹ thuật theo chiều sâu, nâng cấp nhận thức về kỹ thuật, đồng thời các feature package đều có ít nhất 1 người phát triển/bảo trì.
- Khi cần triển khai, phân tán code mới đi các project đã triển khai được nhanh chóng và đơn giản với package manager, CI/CD
- Dễ dàng chia nhỏ để test, build và test liên tục
- Chặt cửa sổ vỡ 1 cách đơn giản nhất

## Nhược điểm:

- Chất lượng code có thể không đồng đều giữa các package
- Việc phân mảng service có thể gây lặp code nếu tech lead hoặc senior không có phương án tổ chức interface hợp lý



# Codebase Topology





## Điều gì mới thực sự “make sense”

Hướng developer tập trung vào giá trị tạo ra cho sản phẩm bằng việc khuyến khích anh em sử dụng năng lực công nghệ và tìm tòi để làm sản phẩm, không phải chỉ là “thợ code” làm đi làm lại những thứ không có “challenges”



# Những yếu tố làm nên giá trị

**Giá trị:** Bạn đóng góp bao nhiêu vào Product goal?

**Chất lượng:** Đầu ra của bạn là sản phẩm ăn xổi hay có giá trị bền vững?

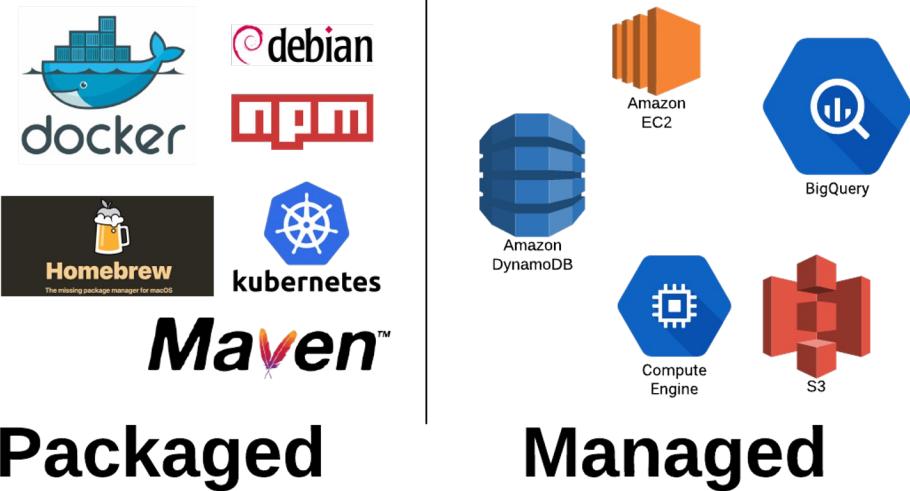
**Teamwork:** Bạn làm những người xung quanh tốt lên bao nhiêu?

# Mẫu chốt của cách làm mới

---

Nếu nhìn vào mô hình bên trên, các ưu nhược điểm thì dễ thấy việc triển khai 1 codebase hiện tại có 1 số điểm cần lưu ý như sau:

- Sử dụng Package Manager
- Chia nhỏ theo mô hình Micro services
- Tích hợp, phát triển và bảo trì liên tục
- Tài liệu: đơn giản, rõ ràng, dễ hiểu, dễ tích hợp
- Maintainer: Đảm bảo ít nhất 1 dev hiểu package và dev rất hiểu package, có tình yêu với nó
- Codebase luôn được nâng cấp, hoàn thiện, và thích hợp với nhiều version, nền tảng, hệ điều hành





## QUAN TRỌNG

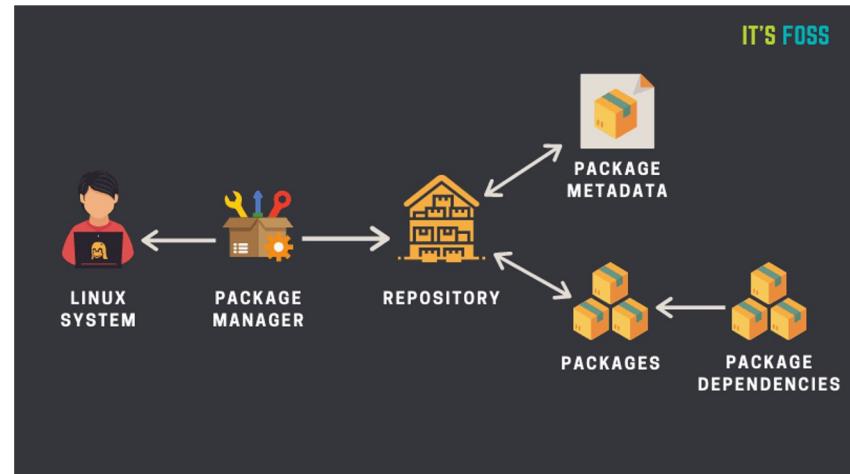
Mọi người có cơ hội sửa sai, cải tiến những đoạn code, những gói mình đã viết ra bất cứ lúc nào, kể cả là khi dự án đã kết thúc và không còn cần phải sửa / nâng cấp nữa.

Qua đó làm mới / tốt hơn mỗi ngày, từ đó mà càng về sau, yếu tố kinh nghiệm, gắn bó, rèn luyện kỹ năng của dev ngày càng được nâng cao.

## Các thành phần của Package Manager (PM), nguyên tắc hoạt động

Thông thường, một hệ thống PM thường bao gồm các thành phần:

- 1. Package Manager:** Cài đặt trên máy developer, quản lý việc cài đặt các package
- 2. Repository:** Nơi chứa các package (trên mạng). Khi cần một package nào đó, PM sẽ tải package đó từ repository về
- 3. Local Package Database:** Mỗi dự án sẽ có local package database riêng, chứa thông tin (metadata, bao gồm *tên package, phiên bản, dependency*) của các package trong dự án đó.



# PHP Package Manager

Lấy ví dụ như PHP, 3 thành phần trên sẽ lần lượt là:

- Package Manager:** composer. Đây là command line tool để quản lý package đi kèm với php
- Repository:** packagist.org - Nơi chứa toàn bộ các package PHP.
- Local Package Database:** Mỗi dự án PHP sẽ có một file *composer.json*, chứa toàn bộ thông tin về phiên bản của các package.

```
{ "type": "library", "name": "beetsfwbs/bs-base-api-lite", "description": "Base Backend API use PHP by Beetssoft WebSystem Team", "keywords": [ "backend", "helper", "library", "php" ], "homepage": "https://github.com/beetsfwbs/bs-base-api-lite", "license": "MIT", "minimum-stability": "stable", "authors": [ { "name": "Nguyen An Hung", "email": "nguyenanhung@egnyeanhung.com", "homepage": "https://www.nguyenanhung.com", "role": "Developer" } ], "require": { "php": ">=7.1.8", "ext-ctype": "*", "ext-curl": "*", "ext-dbase": "*\\n", "ext-dom": "*\\n", "ext-mbstring": "*\\n", "ext-zip": "*\\n", "egnyeanhung/my-debug": "v3.0.*", "v3.0.0", "egnyeanhung/my-cache": "v3.0.*", "v3.0.0", "egnyeanhung/my-logger": "v3.0.*", "v3.0.0", "egnyeanhung/database": "v3.0.*", "v3.0.8.4", "egnyeanhung/monitor": "v3.0.*", "v3.0.5", "egnyeanhung/helpers": "v3.0.*", "v3.0.5", "egnyeanhung/validator": "v3.0.*", "v3.0.5", "egnyeanhung/validator": "v3.0.*", "v3.0.0", "league/oauth2-jwt": "v4.1 || v4.0 || v3.4.0*, v3.5", "league/oauth2-google": "v4.0 || v3.0 || v2.0*, v4.0", "league/oauth2-facebook": "v4.0 || v3.0 || v2.0*, v4.0", "league/oauth2-instagram": "v3.0 || v2.0 || v1.0*, v1.0" }, "require-dev": { "egnyeanhung/bs-base-api": "v1.0.*", "v1.0.0" } }
```

# Phân biệt giữa Feature Package và Project Base

## Feature Package

- Các package được minimal feature, quan điểm mỗi 1 feature (user, roles, permission...) sẽ được đóng gói thành 1 package với đầy đủ các method
- Store trên Github
- Đánh version rõ ràng theo semantic versioning

## Project Base

- Project base là lớp ngoài, điều khiển và lắp ráp các feature package, control chúng để lấy dữ liệu, trả về nơi thích hợp
- Có thể store trên bất cứ đâu: github, gitlab, bitbucket hay backlog, private git server
- Đánh version theo release và milestone

# Các lưu ý về việc đóng gói

- Feature Package và Base Project là 2 khái niệm riêng biệt
- Cần có cách đóng gói public và private package để đảm bảo các gói mang tính chất độc quyền, bảo mật được phân quyền hợp lý
- 1 Package sẽ bao gồm: code, unit test, document... Và đều nên được review chéo, tích hợp thử

# 1 số Package Manager tiêu biểu

---

Đối với các ngôn ngữ lập trình khác, phần lớn đều có Packager, ví dụ

**JS** | NPM / YARN

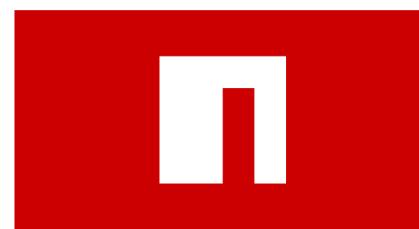
**Ruby** | RubyGems

**Python** | Pipenv, Pipfile, Poetry, Pyflow

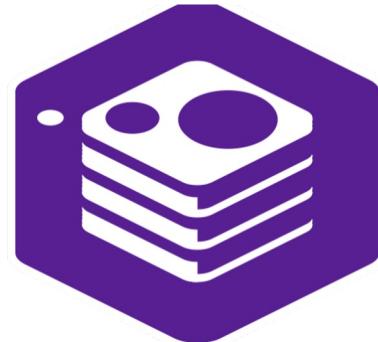
**.NET** | NuGet

**Java** | Maven Central

**GO** | Go Pkg



RubyGems



# Tư duy mẫu chốt của bộ codebase



0  
1

## Quan niệm mỗi feature là 1 service (package)

Mỗi feature/module được hiểu là 1 service riêng. Được đóng gói, lưu trữ ở các repository riêng, phân nhánh, chia tags và đầy đủ wiki, readme, description

## Chia để trị

Khi các feature được chia nhỏ có thể giao cho từng người, với document đầy đủ và có người chịu trách nhiệm về package thì việc bàn giao khi có release nhân sự cũng dễ dàng hơn. Đồng thời đảm bảo trong 1 khung thời gian có ít nhất 1 người hiểu về service đó



0  
2



0  
3

## Tuân chuẩn

Các package được đóng gói, tuân 1 chuẩn được team viết ra (interface, coding convention). Đồng thời giao tiếp với nhau thông qua chuẩn phổ biến: API, static, new method ...

# Duy trì tài liệu

Để kiến trúc codebase dễ tiếp cận cần 1 hệ thống tài liệu đủ đơn giản, rõ ràng và mạch lạc

1. Tại base project: cần có được function list, topology để người mới vào hiểu được kiến trúc hệ thống
2. Tại các feature package: cần có tài liệu về phương án tích hợp, input/output, mô tả package, hướng dẫn cài đặt, sử dụng, các thuật ngữ sử dụng
3. Tech stack của codebase
4. Cần có code mẫu tích hợp
5. Cần có thông tin của người phụ trách package
6. Tài liệu được review chéo bởi các thành viên trong team, được duyệt bởi tech lead
7. Maintainer có trách nhiệm cập nhật tài liệu thường xuyên mỗi khi có thay đổi code, versioning -> có thể tận dụng github releases hoặc github wiki
8. Đối với API: Khuyến nghị sử dụng postman để rõ input/output/endpoint

## BS-BaseAPI-Laravel

### Base API Laravel



#### Bắt đầu

Bộ Base API Laravel này sử dụng package `beetssoftweb` từ repository `bs-base-api-lite` implement vào framework laravel để viết. Base API là một API RESTful dựa trên các request và trả về response dưới dạng JSON. Bộ document được viết ở postman kèm các api mẫu giúp người dùng dễ dàng kết nối và sử dụng.

#### Register user (IMPORTANT)

- Khi đăng ký người dùng sẽ được cấp 1 `username (client_id)` và `secretkey`.
- `client_id` và `secret_key` được dùng để mã hóa thông tin khi gửi request xác thực thông tin người gửi lên hệ thống.

#### Các biến và response trả về

##### 1. Variables postman

Các biến dùng chung của các api thì được đặt trong variables của collection

Ví dụ:

##### 2. Chi tiết một api

Ngoài các input truyền lên, 1 params của api sẽ cần truyền thêm `username` (tức `client_id` được cấp và `signature`).

Signature truyền lên sử dụng thuật toán md5 mã hóa chuỗi các value của từng api quy định, nối với nhau bằng ký tự `PREFIX_AUTH=$` với `client_id`, `secret_key`

Ví dụ một signature của api config được tạo như sau:

Api yêu cầu signature truyền lên là mã hóa chuỗi gồm id, language, client\_id, secret\_key nối với nhau bởi `PREFIX_AUTH` sau:

JUMP TO
Introduction
> Category
> Auth
> Config
> User
> Option
> Tag
> Topic
> DepartmentStructure
> Permission
> User Group
> Menu
> Link
> Post
> Banner
> Language
> ProbeOpinion
> Question
> Source

# Kiến trúc PHP Package mẫu - Dùng build API Package

📁 .github/workflows	Create Project	3 months ago
📁 docs	Create Project	3 months ago
📁 helpers	Create Project	3 months ago
📁 src	Update vendor	2 months ago
📁 test	Update Test	3 months ago
📁 tmp	Create Project	3 months ago
📄 .gitignore	Create Project	3 months ago
📄 CHANGELOG.md	Update vendor	2 months ago
📄 CONTRIBUTING.md	Create Project	3 months ago
📄 LICENSE	Create Project	3 months ago
📄 README.md	Update readme	2 months ago
📄 composer.json	Update vendor	2 months ago

📁 README.md edit

stable v1.0.4 | downloads 1 | unstable dev-main | license MIT | php >=7.1.3

## Template start Backend API Package

Template for repository Backend API - Basic, Simple and Lightweight

[Use this template](#)

api php jwt rest backend  
service

Readme  
MIT license  
1 star  
2 watching  
1 fork

Releases 5

Release version 1.0.4 Latest  
on Aug 7  
+ 4 releases

Packages

No packages published  
[Publish your first package](#)

Languages

PHP 100.0%

templates	Init Project	3 months ago
test	Add test	3 months ago
tmp	Init Project	3 months ago
.gitignore	Init Project	3 months ago
CHANGELOG.md	add method	2 months ago
CONTRIBUTING.md	Init Project	3 months ago
LICENSE	Init Project	3 months ago
README.md	Update vendor	2 months ago
composer.json	Update vendor	2 months ago

☰ README.md



stable v1.0.3 | downloads 1 | unstable dev-main | license MIT | php >=7.1.3

## Releases 4

 Release version 1.0.3 Latest  
on Aug 7

+ 3 releases

## Packages

No packages published  
[Publish your first package](#)

## Languages

Kiến trúc PHP Package mẫu - Dùng build  
Frontend module

main ▾ 1 branch 4 tags Go to file Add file ▾ Code ▾ Use this template

hungnguyenhp add method 78df6a4 on Aug 7 6 commits

.github/workflows	Init Project	3 months ago
docs	Init Project	3 months ago
helpers	Init Project	3 months ago
src	Upgrade database	2 months ago
templates	Init Project	3 months ago
test	Add test	3 months ago
tmp	Init Project	3 months ago
.gitignore	Init Project	3 months ago
CHANGELOG.md	add method	2 months ago
CONTRIBUTING.md	Init Project	3 months ago
LICENSE	Init Project	3 months ago
README.md	Update vendor	2 months ago
composer.json	Update vendor	2 months ago

README.md

stable v1.0.3 downloads 1 unstable dev-main license MIT php >=7.1.3

## Template start Frontend Module Package

Template for repository Frontend Module - Basic, Simple and Lightweight

### Use this Template

First, you can [Use this template](#) for new project: [Use this template](#)

Second, clone your project to your path in your machine

About

Structure Repository Template for Frontend Module Service Package

Readme

MIT license

1 star

2 watching

0 forks

---

Releases 4

Release version 1.0.3 (Latest) on Aug 7 + 3 releases

---

Packages

No packages published [Publish your first package](#)

---

Languages

PHP 100.0%

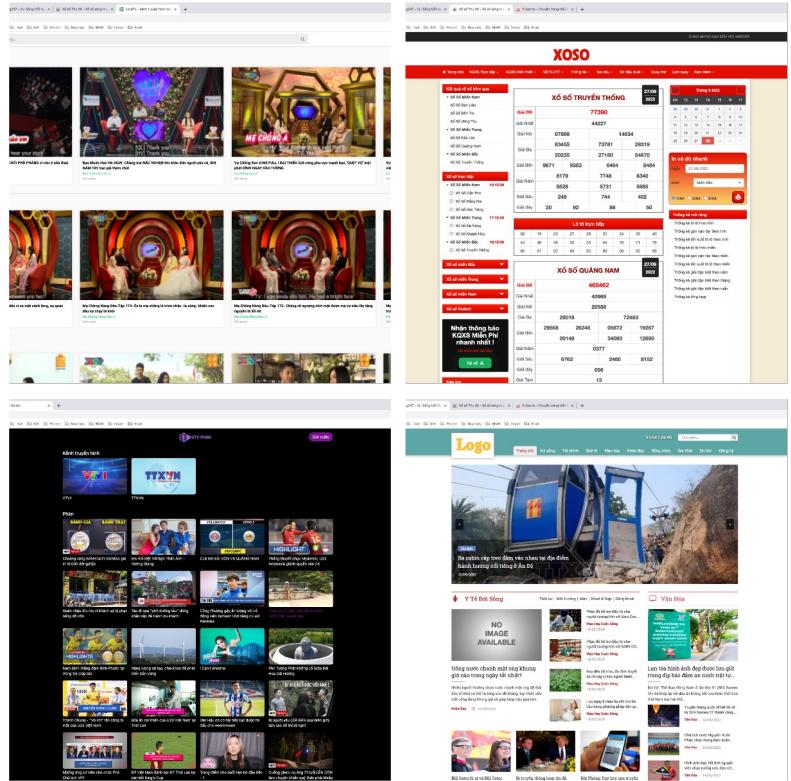
Kiến trúc PHP Package mẫu - Dùng build Frontend module

# Ví dụ về 1 số product build từ chung 1 codebase với triết lý trên

- Chung 1 codebase với khoảng 70% package dùng chung
- Các feature riêng vẫn được đóng gói để có thể tái sử dụng (data, logic)

4 website bên cạnh cùng 1 codebase nhưng có thể triển khai những hạng mục product khác nhau: video streaming, movies, news portal, xổ số

Thực tế cũng đã triển khai 1 số APIs, tuy nhiên APIs không dễ show nên không capture tại đây



---

# Our member

Đội ngũ triển khai  
codebase team  
WebSystem



PHP Advisor

Nguyễn An Hùng

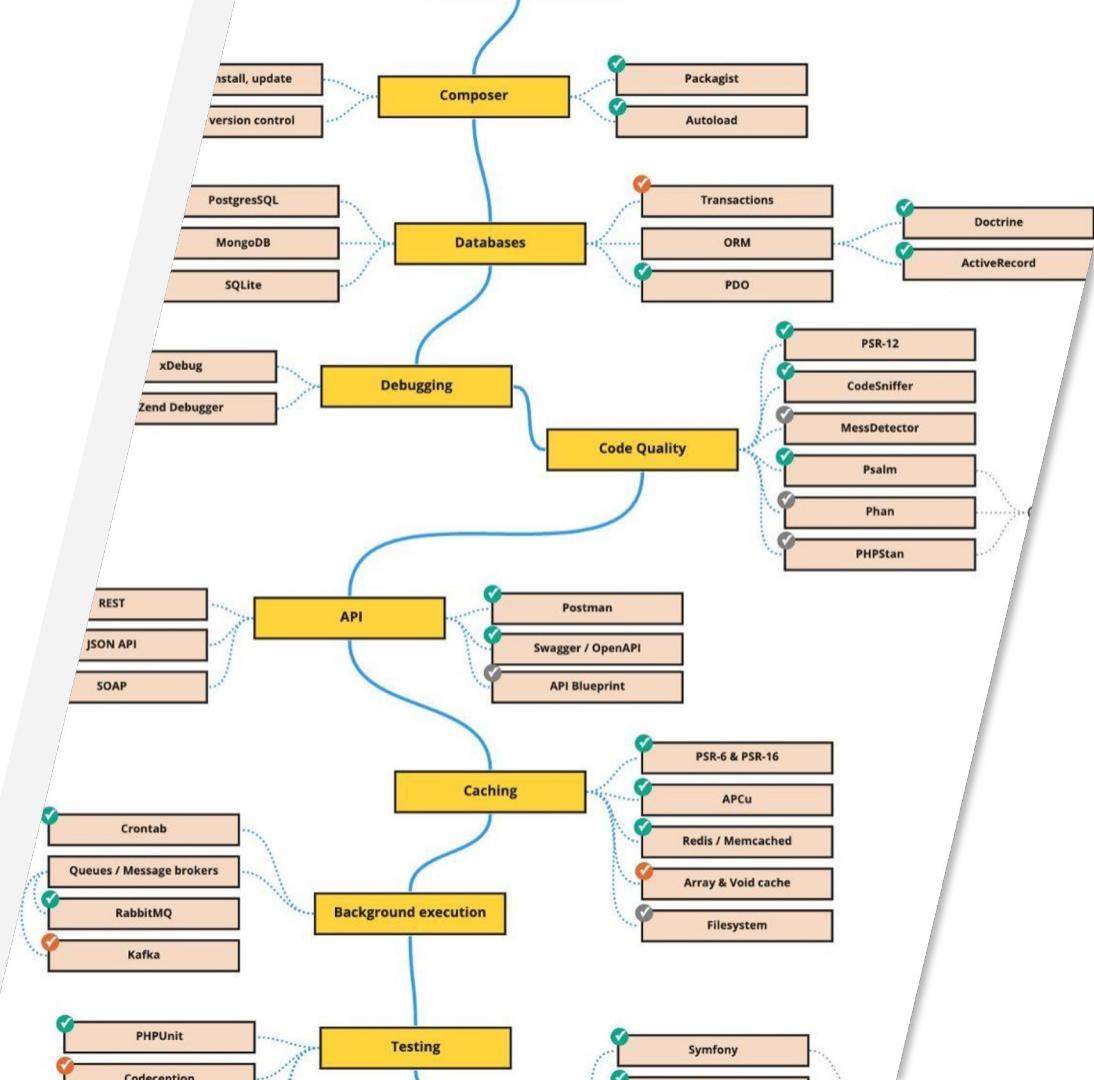


PHP Developer

Nhâm Quang Huy

# Roadmap

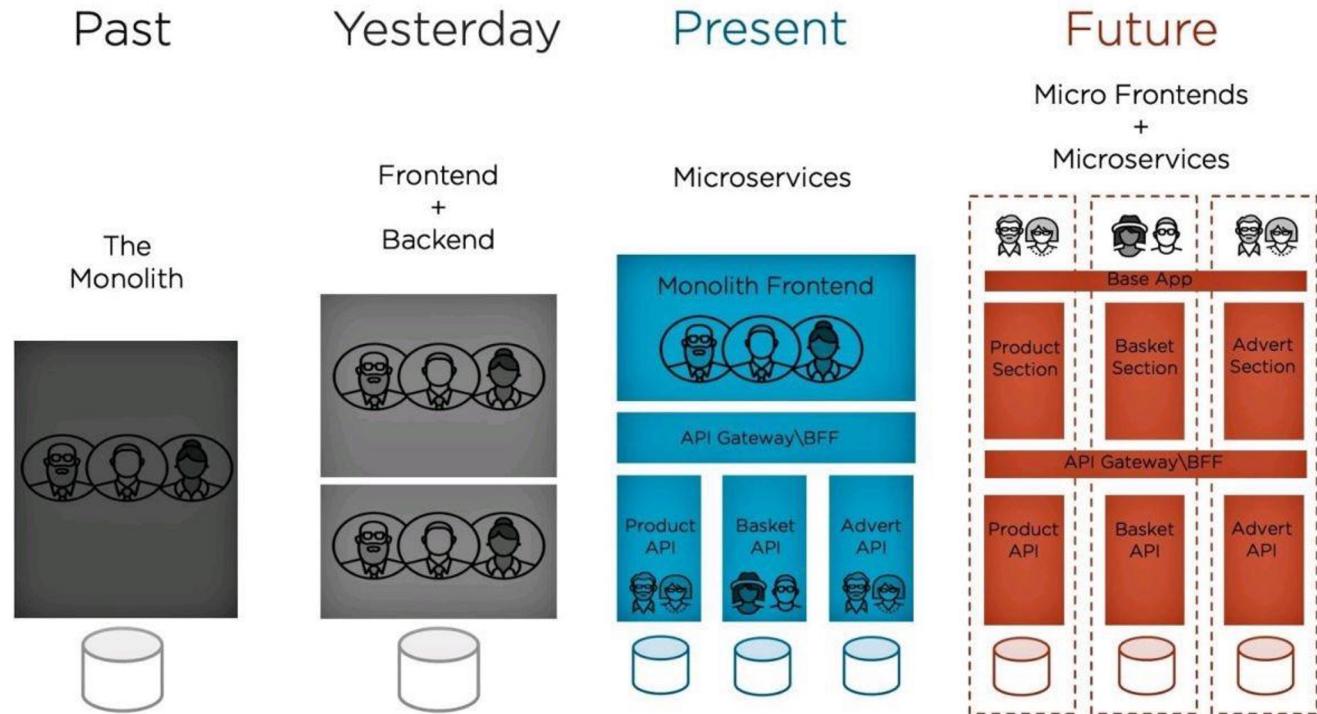
Nhìn vào sơ đồ bên cạnh,  
chúng ta có thể thấy chúng ta  
phải học rất nhiều :)))



# Microservices

## Architect

Giới thiệu về kiến trúc phát triển phần mềm xưa và nay



## Architecture



## Workshop sau có gì?

- Các nguyên tắc và phương pháp khi xây dựng codebase và đóng gói code
- Giới thiệu 1 số cách thức và mindset giúp dev có thể tiếp cận thêm nhiều nguồn tri thức, quan hệ, mentor

# HOMEWORK

Mỗi nhóm cùng nhau xây dựng 1 package (bằng ngôn ngữ nào cũng được), đáp ứng được các yêu cầu về: code, document, unit test và thuyết trình tại Workshop 15/11/2022.

- Trước **17h30** ngày **10/11** gửi lại tên đề tài,
  - Trước **17h30** ngày **14/11** gửi lại đường link các package trên github.
-



Thanks & QA  
Hẹn gặp các  
chiến hữu ở  
workshop sau