Laravel MVC

Web Programming: Beginner (Integrative-Generative AI Edition)

Author: Student Price Book Center

ai

คำนำ

ในยุคปัจจุบันที่การพัฒนาเว็บแอปพลิเคชันเป็นที่ต้องการอย่างสูงทั้งในระดับองค์กร บริษัท ห้างร้าน และหน่วยงานต่าง ๆ การเลือกใช้เทคโนโลยีที่มีประสิทธิภาพและง่ายต่อการเรียนรู้ถือเป็นปัจจัยสำคัญในการสร้างสรรค์ระบบที่ตอบโจทย์การทำงานจริง **Laravel** จึงกลายเป็นหนึ่งใน PHP Framework ที่ได้รับความนิยมสูงสุด ด้วยความโดดเด่นในเรื่องโครงสร้างที่ชัดเจนตามแนวคิด **MVC (Model-View-Controller)**, ความสามารถในการพัฒนาอย่างรวดเร็ว, และการมีเครื่องมืออำนวยความสะดวกที่ครบถ้วน ทำให้ Laravel เป็นตัวเลือกที่เหมาะสมสำหรับทั้งผู้เริ่มต้นและนักพัฒนามืออาชีพ

หนังสือ **Laravel MVC Web Programming: Beginner** เล่มนี้ ถูกจัดทำขึ้นเพื่อตอบสนองความต้องการของผู้ที่ต้องการเริ่มต้นพัฒนาเว็บแอปพลิเคชันด้วย Laravel อย่างเป็นระบบและครบถ้วน โดยเนื้อหาได้รับการออกแบบอย่างละเอียด ไล่เรียงตามลำดับขั้นตอนตั้งแต่พื้นฐานจนถึงการสร้างโปรเจกต์จริง เพื่อให้ผู้อ่านเข้าใจทั้งแนวคิดและการประยุกต์ใช้งานจริง

ใน **บทที่ 1** ผู้อ่านจะได้เรียนรู้เกี่ยวกับ Laravel ตั้งแต่ประวัติ ความเป็นมา วิวัฒนาการ และแนวโน้มการใช้งานในอนาคต ตลอดจนขั้นตอนการติดตั้งและการตั้งค่าเริ่มต้นอย่างละเอียด ผ่านเครื่องมือที่แนะนำ เช่น **Visual Studio Code (VS Code)** และ **NetBeans** พร้อมการทดสอบโปรแกรม Laravel ครั้งแรก ซึ่งจะปูพื้นฐานสำคัญสำหรับการเรียนรู้ในบทถัดไป

บทที่ 2 จะนำผู้อ่านเข้าสู่หัวใจของการพัฒนาเว็บแอปพลิเคชัน นั่นคือการกำหนดเส้นทางหรือ **Routing** และการสร้าง **Controller** เพื่อเชื่อมโยงการทำงานของระบบ ผู้อ่านจะได้ฝึกสร้าง Route ในรูปแบบต่าง ๆ เช่น GET, POST, PUT, และ DELETE รวมถึงการส่งข้อมูลผ่าน **Route Parameters** และ **Query Strings** การตั้งชื่อ Route และการจัดกลุ่ม Route ให้มีความเป็นระเบียบ

บทที่ 3 จะอธิบายการจัดการ **View และ Blade Templating** ซึ่งเป็นเครื่องมือที่ช่วยให้การสร้างหน้าเว็บเป็นเรื่องง่ายและเป็นระบบ โดยเนื้อหาครอบคลุมตั้งแต่การสร้าง Layout ด้วย Blade Template Engine, การใช้คำสั่ง Blade Directives อย่าง if และ loop, การแสดงข้อมูลที่ Controller ส่งมายัง View ไปจนถึงตัวอย่างโปรเจกต์จริงที่ช่วยให้เข้าใจภาพรวมได้ดียิ่งขึ้น

ใน **บทที่ 4** ผู้อ่านจะได้เรียนรู้การจัดการฟอร์ม (**Form**) และการตรวจสอบข้อมูล (**Validation**) ซึ่งเป็นส่วนสำคัญของทุกเว็บแอปพลิเคชัน โดยเนื้อหาจะสอนวิธีสร้างฟอร์มด้วย Blade, การรับข้อมูลผ่าน Request Object, การใช้เมธอด validate ใน Controller และการแสดงผลข้อผิดพลาดเพื่อสร้างประสบการณ์ผู้ใช้ที่ดีและปลอดภัย พร้อมตัวอย่างโปรเจกต์ที่ประยุกต์ใช้งานได้จริง

หนังสือเล่มนี้ยังให้ความสำคัญกับการสอนแบบ “ลงมือทำ” โดยเนื้อหาถูกออกแบบให้มี **ตัวอย่างโค้ดจริง** อธิบายอย่างละเอียดทุกบรรทัด พร้อม **ตัวอย่างโปรเจกต์บูรณาการ** ที่สอดคล้องกับการใช้งานในสถานการณ์จริง เพื่อให้ผู้อ่านสามารถนำไปพัฒนาต่อยอดได้ทันที

ผู้เขียนหวังเป็นอย่างยิ่งว่าหนังสือเล่มนี้จะเป็นคู่มือที่ช่วยให้ผู้อ่านเข้าใจการทำงานของ Laravel ได้อย่างเป็นลำดับขั้น มีพื้นฐานที่มั่นคงในการพัฒนาเว็บแอปพลิเคชันเชิงโครงสร้าง และสามารถก้าวต่อไปสู่ระดับที่สูงขึ้นได้อย่างมั่นใจ

สุดท้ายนี้ ผู้เขียนขอขอบคุณทีมงานและผู้มีส่วนเกี่ยวข้องทุกท่านที่ให้คำปรึกษาและสนับสนุนการจัดทำหนังสือเล่มนี้ และหวังเป็นอย่างยิ่งว่าหนังสือเล่มนี้จะเป็นประโยชน์ต่อผู้อ่านทุกท่านในการเรียนรู้ Laravel และสร้างสรรค์ผลงานคุณภาพต่อไปในอนาคต

ด้วยรักและปรารถนาดี

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1
ทำความรู้จัก Laravel และติดตั้งเบื้องต้น
(Introduction to Laravel Framework)

เนื้อหา

- ทำความรู้จัก Laravel และการติดตั้งเบื้องต้น
- ทำความรู้จัก Laravel และติดตั้งเบื้องต้นเพิ่มเติม
- ประวัติ วิวัฒนาการ การใช้งานในปัจจุบัน และแนวโน้มในอนาคตของ Laravel
- องค์กร บริษัท ห้างร้าน และหน่วยงานต่าง ๆ ที่ใช้ Laravel
- Laravel คืออะไร, จุดเด่น และแนวคิด MVC
- ขั้นตอนการติดตั้ง Laravel ผ่าน Composer
- เครื่องมือ IDE / Code Editor ที่แนะนำสำหรับ Laravel
- การติดตั้ง Laravel บน VS Code และ NetBeans พร้อม วิธีทดสอบการเขียนโปรแกรม Laravel อย่างละเอียด
- โครงสร้างโฟลเดอร์โปรเจกต์ Laravel
- การตั้งค่า Environment ใน Laravel ผ่านไฟล์ .env
- การรันโปรเจกต์ Laravel ด้วยคำสั่ง php artisan serve
- การพัฒนาโปรเจกต์ Laravel แรก ด้วย VS Code และ NetBeans

บทนำ

บทที่ 1: ทำความรู้จัก Laravel และติดตั้งเบื้องต้น

Laravel เป็นหนึ่งในเฟรมเวิร์ก PHP ที่ได้รับความนิยมสูงสุดในปัจจุบัน ด้วยจุดเด่นด้านความง่ายในการพัฒนาเว็บแอปพลิเคชัน โครงสร้างที่เป็นระเบียบ และมีเครื่องมืออำนวยความสะดวกหลากหลาย ทำให้นักพัฒนาทั้งมือใหม่และมืออาชีพเลือกใช้ Laravel อย่างแพร่หลาย การออกแบบที่ทันสมัยของเฟรมเวิร์กนี้ยังรองรับการขยายระบบในอนาคต และช่วยลดความซับซ้อนของการพัฒนาได้อย่างมีประสิทธิภาพ หนึ่งในเหตุผลสำคัญที่ทำให้ Laravel เป็นที่นิยม คือการยึดตามแนวคิด **MVC (Model-View-Controller)** ซึ่งเป็นสถาปัตยกรรมที่แยกการทำงานของส่วนข้อมูล (Model) ส่วนการแสดงผล (View) และส่วนควบคุมการประมวลผล (Controller) ออกจากกันอย่างชัดเจน แนวคิดนี้ช่วยให้นักพัฒนาสามารถเขียนโค้ดที่เป็นระบบ ง่ายต่อการบำรุงรักษา และสามารถทำงานร่วมกันในทีมได้อย่างมีประสิทธิภาพ

เพื่อเริ่มต้นการใช้งาน Laravel ขั้นตอนแรกที่จำเป็นคือการติดตั้งเฟรมเวิร์กผ่าน **Composer** ซึ่งเป็นเครื่องมือจัดการแพ็กเกจมาตรฐานของ PHP Composer ช่วยให้การติดตั้ง Laravel และไลบรารีที่เกี่ยวข้องทำได้อย่างสะดวก รวดเร็ว และมีการจัดการเวอร์ชันของไลบรารีให้สอดคล้องกันโดยอัตโนมัติ ทำให้มั่นใจได้ว่าการพัฒนาโปรเจกต์จะดำเนินไปอย่างราบรื่น หลังจากติดตั้ง Laravel แล้ว การทำความเข้าใจโครงสร้างโฟลเดอร์ของโปรเจกต์ถือเป็นขั้นตอนสำคัญ เนื่องจาก Laravel มีการจัดระเบียบไฟล์และโฟลเดอร์อย่างชัดเจน แต่ละส่วนถูกออกแบบให้มีหน้าที่เฉพาะ เช่น โฟลเดอร์ app ใช้สำหรับเขียนโค้ดธุรกิจหลัก, โฟลเดอร์ resources สำหรับเก็บไฟล์วิวและไฟล์ทรัพยากรต่างๆ, และโฟลเดอร์ routes สำหรับกำหนดเส้นทางของแอปพลิเคชัน ความเข้าใจในโครงสร้างนี้จะช่วยให้นักพัฒนาทำงานได้อย่างเป็นระบบและลดข้อผิดพลาดที่อาจเกิดขึ้น

อีกส่วนสำคัญที่ต้องตั้งค่าให้ถูกต้องคือไฟล์ **Environment (.env)** ซึ่งเป็นไฟล์ที่ใช้สำหรับกำหนดค่าต่างๆ ของโปรเจกต์ เช่น การเชื่อมต่อฐานข้อมูล, การตั้งค่า URL หลัก, หรือคีย์ลับสำหรับการเข้ารหัส การตั้งค่าในไฟล์นี้ทำให้การจัดการสภาพแวดล้อมของระบบ เช่น การพัฒนาหรือการใช้งานจริง มีความยืดหยุ่นและปลอดภัยมากขึ้น เมื่อเสร็จสิ้นการตั้งค่าพื้นฐานทั้งหมดแล้ว นักพัฒนาสามารถเริ่มต้นรันโปรเจกต์ Laravel ได้ทันทีโดยใช้คำสั่ง **Artisan serve** ซึ่งเป็นคำสั่งภายในของ Laravel สำหรับเปิดเซิร์ฟเวอร์ทดสอบแบบชั่วคราว คำสั่งนี้ช่วยให้ผู้พัฒนาสามารถตรวจสอบผลลัพธ์ของการพัฒนาได้ทันทีผ่านเว็บเบราว์เซอร์ และเป็นขั้นตอนสำคัญก่อนการปรับใช้จริงบนเซิร์ฟเวอร์โปรดักชัน

บทนี้จึงถือเป็นก้าวแรกที่สำคัญสำหรับผู้ที่ต้องการเรียนรู้การพัฒนาเว็บแอปพลิเคชันด้วย Laravel เพราะจะปูพื้นฐานความเข้าใจเกี่ยวกับแนวคิดหลักของเฟรมเวิร์ก การติดตั้ง การจัดการโครงสร้างโปรเจกต์ และการตั้งค่าเบื้องต้นอย่างครบถ้วน ความเข้าใจในขั้นตอนเหล่านี้จะเป็นรากฐานที่มั่นคงสำหรับการเรียนรู้หัวข้อที่ซับซ้อนยิ่งขึ้นในบทต่อๆ ไป

ทำความรู้จัก Laravel และการติดตั้งเบื้องต้น

□ 1. Laravel คืออะไร, จุดเด่น และแนวคิด MVC

Laravel เป็น Web Framework ของภาษา **PHP** ที่พัฒนาโดย **Taylor Otwell** โดยมุ่งเน้นให้การพัฒนาเว็บแอปพลิเคชันเป็นไปอย่างมีโครงสร้าง สะดวก ปลอดภัย และสามารถขยายระบบได้ในระดับองค์กร

จุดเด่นของ Laravel:

- □ Syntax ที่อ่านง่ายและ "เป็นธรรมชาติ"
- □ รองรับ **MVC** (Model-View-Controller)
- □ มี ORM ชื่อ **Eloquent** สำหรับทำงานกับฐานข้อมูล
- □ มี **Routing**, **Middleware**, **Queue**, **Authentication**, **Notification**, **Mail** และอีกมากมาย
- □ รองรับระบบ **Testing**, **Queue**, **Event Broadcasting** ฯลฯ

แนวคิด MVC:

Laravel ยึดหลักโครงสร้าง **MVC** เพื่อแยกหน้าที่ของแต่ละส่วน:

- **Model (M)** ⟶ เชื่อมต่อกับฐานข้อมูล / ธุรกิจ
- **View (V)** ⟶ ส่วนแสดงผลหน้าจอ (HTML/Blade)
- **Controller (C)** ⟶ ควบคุมตรรกะการไหลของข้อมูลระหว่าง M และ V

□ 2. การติดตั้ง Laravel ผ่าน Composer

2.1 ติดตั้ง Composer (ถ้ายังไม่มี)

Laravel ต้องใช้ **Composer** ซึ่งเป็น PHP Dependency Manager

- สำหรับ Windows: โหลดจาก https://getcomposer.org
- ตรวจสอบว่า composer ติดตั้งสำเร็จ:

```
composer -V
```

2.2 สร้างโปรเจกต์ Laravel ใหม่:

```
composer create-project laravel/laravel myproject
```

หรือใช้ Laravel Installer:

```
composer global require laravel/installer
laravel new myproject
```

2.3 เข้าสู่โฟลเดอร์โปรเจกต์:

```
cd myproject
```

□ 3. โครงสร้างโฟลเดอร์โปรเจกต์ Laravel

เมื่อสร้างโปรเจกต์เสร็จ จะมีโครงสร้างโฟลเดอร์ประมาณนี้:

```
myproject/
│
├── app/            ⟶ Logic ของระบบ (Models, Controllers, Services)
├── bootstrap/      ⟶ Bootstrap Application
├── config/         ⟶ การตั้งค่าต่าง ๆ
├── database/       ⟶ Migrations, Seeders
├── public/         ⟶ จุดเริ่มต้นของเว็บ (index.php)
├── resources/      ⟶ View (Blade), Language Files
├── routes/         ⟶ routes/web.php, api.php
├── storage/        ⟶ Logs, Cache, Files
├── tests/          ⟶ Unit / Feature Tests
└── .env            ⟶ ไฟล์ตั้งค่าระบบเฉพาะแต่ละเครื่อง
```

☐ 4. การตั้งค่า Environment (.env)

Laravel ใช้ไฟล์ .env สำหรับตั้งค่าระบบแยกตามเครื่องหรือสภาพแวดล้อม เช่น dev, staging, production

ตัวอย่างค่าที่ใช้บ่อย:

APP_NAME=Laravel

APP_ENV=local

APP_KEY=base64:...

APP_DEBUG=true

APP_URL=http://localhost

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=laravel_db

DB_USERNAME=root

DB_PASSWORD=

☐ ค่าพวกนี้จะถูกเรียกใช้ในโค้ดผ่านฟังก์ชัน env() หรือ config()

☐ 5. การรันโปรเจกต์ด้วย Artisan serve

Laravel มีเครื่องมือ CLI ชื่อว่า **Artisan** ที่ช่วยควบคุมระบบหลายอย่าง เช่น สร้าง controller, model, migration และรันเว็บเซิร์ฟเวอร์

คำสั่งรันเซิร์ฟเวอร์:

php artisan serve

ระบบจะเริ่มรันที่:

http://127.0.0.1:8000

☐ ถ้าไม่มี error จะสามารถเปิดเบราว์เซอร์แล้วเห็นหน้า Welcome ของ Laravel

---

☐ สรุปบทนี้

หัวข้อ	สาระสำคัญ
Laravel คือ	PHP Framework ที่เน้นความเรียบง่าย มีโครงสร้าง และมีความสามารถสูง
MVC	แยกหน้าที่ Model, View, Controller ชัดเจน
การติดตั้ง	ใช้ Composer ในการสร้างโปรเจกต์

หัวข้อ	สาระสำคัญ
โฟลเดอร์	ถูกออกแบบให้แยกความรับผิดชอบชัดเจน
.env	ใช้จัดการค่าระบบที่เปลี่ยนได้ตามเครื่อง
Artisan	CLI ที่ใช้รันระบบและจัดการคำสั่งต่าง ๆ

ทำความรู้จัก Laravel และติดตั้งเบื้องต้นเพิ่มเติม

☐ **หัวข้อที่ 1: Laravel คืออะไร, จุดเด่น และแนวคิด MVC**

☐ **Laravel คือ**

Laravel คือ **PHP Web Framework แบบ Full-Stack** ที่ใช้แนวคิด MVC (Model-View-Controller) เพื่อแยก concerns ในการพัฒนาเว็บให้ชัดเจน ซึ่งช่วยลดความซับซ้อนของระบบขนาดใหญ่

Laravel ถูกออกแบบมาให้:

- **เป็นมิตรกับนักพัฒนา** (Developer Friendly)
- **เขียนน้อย ทำได้มาก** (Elegant Syntax)
- **มีเครื่องมือที่จำเป็นครบถ้วน** (Batteries Included)

☐ **จุดเด่น Laravel (เชิงลึก)**

จุดเด่น	รายละเอียด
Routing ที่ยืดหยุ่น	รองรับ Method เช่น GET/POST/PUT/DELETE + Middleware
Blade Templating Engine	รองรับ inheritance (@extends), component (@component)
Eloquent ORM	ทำงานกับฐานข้อมูลเชิง Object-Oriented
Migrations + Seeder	จัดการ schema + mock data แบบ version control
Artisan CLI	สร้าง Model, Controller, Migration และงานอัตโนมัติ
Service Container (IoC)	รองรับ Dependency Injection
Service Provider	จุดเริ่มต้นของ component ต่าง ๆ
Event + Listener	รองรับ Event-driven Architecture
Testing	รองรับทั้ง Unit และ Feature Testing ในตัว
Queue / Schedule / Notification / Mail	รองรับงานระบบเบื้องหลังแบบ built-in

□ หัวข้อที่ 2: การติดตั้ง Laravel ผ่าน Composer

□ วิธีติดตั้ง Laravel (อธิบายเชิงลึก)

ตัวเลือก 1: ใช้ create-project

composer create-project laravel/laravel myapp

- โหลด Laravel เวอร์ชันล่าสุด
- ติดตั้ง dependencies ที่จำเป็นใน vendor/
- สร้างโครงสร้างโปรเจกต์พร้อมใช้งาน

ตัวเลือก 2: ใช้ Laravel Installer (เร็วกว่า)

composer global require laravel/installer

laravel new myapp

□ หลังติดตั้ง Laravel แล้ว จำเป็นต้อง **ตั้งค่า PATH ให้กับ global composer bin** เช่น:

export PATH="$HOME/.composer/vendor/bin:$PATH"

ตรวจสอบว่าใช้งานได้:

php artisan --version

□ หัวข้อที่ 3: โครงสร้างโฟลเดอร์ Laravel (อธิบายแต่ละโฟลเดอร์)

โฟลเดอร์	รายละเอียดเชิงลึก
app/	โค้ดหลักของระบบ เช่น Controller, Model, Service, Event
bootstrap/	การ boot ระบบ (เช่น โหลด autoload, config cache)
config/	ไฟล์ .php สำหรับตั้งค่า module ต่าง ๆ เช่น database, mail
database/	มี migrations, seeders, factories สำหรับสร้าง mock data
public/	จุด entry ของแอป (ไฟล์ index.php), เก็บไฟล์ static
resources/	มี blade views, language file, และ assets
routes/	กำหนด web/api routes ใน web.php, api.php, console.php
storage/	เก็บไฟล์, log, cache แยกตาม app, logs, framework
tests/	แยก Feature และ Unit test อย่างชัดเจน
vendor/	Dependency ทั้งหมดที่โหลดจาก Composer

□ Laravel ใช้ **PSR-4 autoloading** โดยคลาสใน app/ จะโหลดอัตโนมัติ

⚙□ หัวข้อที่ 4: การตั้งค่า Environment (.env)

.env คืออะไร?

คือไฟล์ config ที่ใช้เก็บ "ข้อมูลเฉพาะเครื่อง" เช่น database, URL, key โดยใช้กับ Laravel Config system (config/*.php)

ตัวอย่าง:

APP_NAME=Laravel

APP_ENV=local

APP_KEY=base64:o9IoO0Z...

APP_DEBUG=true

APP_URL=http://localhost

DB_CONNECTION=mysql

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=my_db

DB_USERNAME=root

DB_PASSWORD=secret

เบื้องหลัง:

- Laravel ใช้ **Dotenv** package ในการโหลดไฟล์ .env
- สามารถใช้ env() ใน config ได้เท่านั้น เช่น:

// config/app.php

'name' => env('APP_NAME', 'Laravel'),

□ อย่าใช้ env() ใน production code เช่น controller หรือ service

---

▶□ หัวข้อที่ 5: การรันโปรเจกต์ด้วย Artisan (php artisan serve)

Artisan คืออะไร?

Artisan คือ CLI tool ของ Laravel ช่วยจัดการคำสั่งต่าง ๆ เช่น

คำสั่ง	ความหมาย
php artisan serve	รันเซิร์ฟเวอร์ local ที่ http://127.0.0.1:8000
php artisan make:controller	สร้าง Controller
php artisan make:model	สร้าง Model
php artisan migrate	รัน migration
php artisan config:cache	compile config เพื่อ performance

คำสั่ง	ความหมาย
php artisan route:list	แสดง route ทั้งหมดในระบบ
php artisan tinker	interactive shell สำหรับทดสอบ model / DB

วิธีใช้ serve:

php artisan serve

- ใช้ PHP built-in web server (เฉพาะเครื่อง dev)
- แนะนำให้ใช้ **Valet / Homestead / Docker** สำหรับ production-level

---

☐ **สรุป (ระดับลึก)**

หัวข้อ	Insight
Laravel คือ	Framework ที่ให้เครื่องมือครบ พร้อมใช้ สร้างระบบระดับมืออาชีพ
จุดเด่น	มี Eloquent, Queue, CLI, Testing, Blade, Config, Auth ในตัว
Composer	ใช้ในการติดตั้ง Laravel และ Package เสริมอื่น ๆ
โครงสร้างโฟลเดอร์	แยก logic อย่างชัดเจน รองรับ Scaling ได้ดี
.env	เก็บค่า config เฉพาะ environment ปลอดภัย และปรับง่าย
Artisan	CLI ตัวหลักของ Laravel มีคำสั่งหลากหลายครอบคลุม

ประวัติ วิวัฒนาการ การใช้งานในปัจจุบัน และแนวโน้มในอนาคตของ Laravel

---

☐ **ประวัติของ Laravel**

Laravel เป็น **PHP Web Framework** แบบ **open-source** ที่พัฒนาโดย **Taylor Otwell** เปิดตัวครั้งแรกในปี **2011** โดยมีเป้าหมายเพื่อจัดการกับข้อจำกัดของ Framework ที่นิยมในยุคนั้นอย่าง CodeIgniter เช่น:

- ไม่มี built-in support สำหรับ authentication
- ไม่มี ORM
- ไม่มีระบบ routing ที่ยืดหยุ่น

จุดเด่นแรกเริ่ม

- ใช้แนวคิด MVC (Model–View–Controller)
- มีระบบ routing และ authentication ที่ง่าย
- มี Blade Template Engine

- ใช้ Eloquent ORM ที่ทำงานกับฐานข้อมูลแบบ object-oriented

วิวัฒนาการของ Laravel

Laravel ได้มีการพัฒนาอย่างต่อเนื่อง โดยมีเวอร์ชันใหม่ออกมาแทบทุกปี:

Laravel 1 (2011)

- เปิดตัวครั้งแรก
- ยังไม่มี Composer
- ยังไม่มี Eloquent ORM (ใช้ ActiveRecord)

Laravel 3 (2012)

- เริ่มใช้ Artisan CLI
- เริ่มรองรับแพ็กเกจและ Bundles
- เริ่มมีระบบ Migration

Laravel 4 (2013)

- เปลี่ยนสถาปัตยกรรมครั้งใหญ่
- ใช้ **Composer** สำหรับ dependency management
- ระบบ IoC Container
- Eloquent ORM เวอร์ชันใหม่

Laravel 5.x (2015–2018)

- มีระบบ Middleware, Request Validation, Job Queue, Events
- Blade Template พัฒนามากขึ้น
- Laravel Mix (webpack)

Laravel 6 (2019)

- เริ่มใช้ Semantic Versioning (เวอร์ชันตาม MAJOR.MINOR.PATCH)
- เพิ่ม Laravel UI, Job Batching

Laravel 7–8 (2020–2021)

- เพิ่ม Laravel Jetstream, Livewire, Inertia.js
- ระบบ Component-based มากขึ้น
- Laravel Sanctum & Laravel Breeze สำหรับ Auth

Laravel 9 (2022)

- รองรับ PHP 8 ขึ้นไป
- Bootstrap บน Symfony 6
- New Query Builder Interface

Laravel 10 (2023)

- เวอร์ชัน LTS (Long Term Support)

- เน้น modern PHP features เช่น attribute-based route
- Performance ดีขึ้นมาก

Laravel 11 (2024)

- เรียบง่ายมากขึ้น
- Bootstrap ใหม่หมด: โฟลเดอร์เล็กลง, config เริ่มต้นน้อยลง
- เน้น “Convention Over Configuration”
- Performance และ DX (Developer Experience) ดีเยี่ยม

⚙ การใช้งาน Laravel ในปัจจุบัน

Laravel กลายเป็น PHP Framework ที่นิยมที่สุดในโลก เนื่องจาก:

- ใช้งานง่าย เรียนรู้เร็ว
- มีระบบ Authentication, Routing, Validation, Queue, Mail, Notification ครบ
- รองรับ **SPA (Single Page Application)** ผ่าน **Inertia.js**, **Livewire**
- มี Ecosystem ครบ เช่น:
    - **Laravel Nova** (Admin Panel)
    - **Laravel Forge** (Server Deployment)
    - **Laravel Vapor** (Serverless on AWS)
    - **Laravel Jetstream** / **Breeze** (Auth Scaffolding)
    - **Laravel Pint** (Code Style)
    - **Laravel Sail** (Docker)

ใช้กันอย่างแพร่หลายใน:

- ระบบภายในองค์กร (ERP, HR, CRM)
- e-Commerce
- ระบบจอง
- SaaS platform
- API backend สำหรับ Mobile App

แนวโน้มการใช้งาน Laravel ในอนาคต

Laravel ยังคงมีแนวโน้มการเติบโตที่ดี ด้วยเหตุผลต่อไปนี้:

1. PHP ยังไม่ตาย

แม้มีภาษาใหม่ๆ เกิดขึ้น แต่ PHP ยังมีฐานผู้ใช้มหาศาล โดย Laravel เป็นตัวเลือกแรกๆ

2. Ecosystem เข้มแข็ง

ระบบเสริมของ Laravel เช่น Nova, Forge, Vapor, Envoyer, Mix ทำให้เหมาะกับทั้งนักพัฒนาเดี่ยวและทีมงานระดับองค์กร

☐ 3. การเชื่อมต่อกับ Frontend Framework

Laravel ใช้ได้ดีกับ Vue.js, React, และแม้กระทั่ง Livewire/Inertia สำหรับ SPA

☐ 4. Developer Experience (DX) ที่ยอดเยี่ยม

มี CLI, scaffolding, automation tools ที่ช่วยให้พัฒนาระบบได้รวดเร็ว

☐ 5. รองรับแนวคิดสมัยใหม่

เช่น Clean Architecture, Hexagonal, Domain-Driven Design (DDD)

☐ 6. ตอบโจทย์ DevOps

มี Laravel Sail สำหรับ Docker, Laravel Forge สำหรับ CI/CD บน Server

---

☐ สรุป

ด้าน	รายละเอียด
ผู้พัฒนา	Taylor Otwell
เริ่มต้นปี	2011
ภาษา	PHP
สถาปัตยกรรม	MVC
จุดเด่น	DX ดีเยี่ยม, Ecosystem ครบ, Performance ดี
เหมาะกับ	Web App, API, SaaS, Internal Systems
แนวโน้ม	เติบโตต่อเนื่อง ด้วยเทคโนโลยีทันสมัยและ community ที่แข็งแกร่ง

รายละเอียดเฉพาะ เช่น:

- เปรียบเทียบ Laravel กับ Symfony หรือ Node.js
- ตัวอย่างโปรเจกต์จริง
- การใช้ Laravel ร่วมกับ Vue หรือ React

ด้านล่างคือรายละเอียดเชิงลึกในหัวข้อที่คุณขอไว้:

☐ เปรียบเทียบ Laravel กับ Symfony และ Node.js

1. Laravel vs Symfony (PHP Framework ทั้งคู่)

เกณฑ์	Laravel	Symfony
รูปแบบ	Opinionated, ใช้งานง่าย, มีโครงสร้างชัดเจน	Flexible, Configurable, ใช้งานได้หลากหลาย แต่เรียนรู้ยากกว่า
Learning	ง่ายกว่า เพราะมี default setup และ	ยากกว่า ต้องเรียนรู้มากก่อนใช้งาน

เกณฑ์	Laravel	Symfony
Curve	Artisan CLI	
Community	ใหญ่มาก โดยเฉพาะในกลุ่ม dev มือใหม่ – กลาง	เน้นกลุ่ม enterprise และโปรเจกต์ขนาดใหญ่
Ecosystem	Laravel Forge, Vapor, Nova, Sail ฯลฯ	ไม่มี official ecosystem เท่ากับ Laravel
Performance	ดีมาก โดยเฉพาะ Laravel 9+	ดีกว่าเล็กน้อยในบาง use case (ด้วย Flexibility)
เหมาะกับใคร	Freelancers, Startups, Teams ที่ต้องการ DX ดี	องค์กรใหญ่, ต้องการ custom architecture

สรุป:

- Laravel ดีสำหรับการพัฒนาเร็ว เรียนรู้ไว
- Symfony ดีสำหรับโครงการที่ต้องการความยืดหยุ่นสูงหรือ custom architecture

---

2. Laravel vs Node.js (ต่างภาษา: PHP vs JavaScript)

เกณฑ์	Laravel (PHP)	Node.js (JavaScript)
ภาษา	PHP	JavaScript (Fullstack)
Concurrency	ใช้ Multi-threaded Apache/Nginx	Event Loop แบบ non-blocking
Performance	เร็วพอสมควร โดยเฉพาะบน PHP 8+	เร็วมากเมื่อทำ async-heavy เช่น Chat, Streaming
ใช้ร่วมกับ Frontend	ใช้ Inertia.js, Livewire, Vue/React ได้ดี	มักจับคู่กับ React, Angular, Vue
Package	ใช้ Composer	ใช้ npm (แพ็กเกจเยอะมาก)
Deployment	เหมาะกับ Apache/Nginx	ต้องจัดการ Server, PM2, Nginx หรือ Docker เอง
Community	ใหญ่ในสาย backend	ใหญ่ที่สุดสาย Fullstack และ Startup

สรุป:

- **Laravel** ดีสำหรับ Web Application และ RESTful API
- **Node.js** เหมาะสำหรับ real-time system, microservices, หรือคนที่ต้องการเขียน Fullstack ด้วยภาษาเดียว

⚙ ตัวอย่างโปรเจกต์จริงที่ใช้ Laravel

1. ระบบจองโรงแรม (Hotel Booking System)

- จัดการผู้ใช้, ห้องพัก, การจอง, การชำระเงิน
- ใช้ Laravel + Vue.js
- ใช้ Laravel Sanctum หรือ Passport สำหรับ API Auth

2. ระบบจัดการพนักงาน (Employee Management System)

- ฟีเจอร์: Login, Roles (Admin, HR, Staff), CRUD, PDF export
- ใช้ Laravel Breeze/Jetstream + Inertia.js

3. SaaS Dashboard

- ระบบสมาชิกแบบ Subscription
- ใช้ Laravel Cashier (Stripe), Laravel Nova (Admin)
- ใช้ TailwindCSS + Livewire สำหรับ UI

การใช้ Laravel ร่วมกับ Frontend Framework

Laravel + Vue.js

- ใช้ Laravel Mix สำหรับ build Vue components
- ใช้ Inertia.js เพื่อทำ SPA แบบไม่มี API layer
- รองรับ SSR (Server Side Rendering) ได้

```
composer require inertiajs/inertia-laravel
npm install @inertiajs/inertia @inertiajs/inertia-vue3 vue@3
```

Laravel + React

- ใช้ Laravel Mix build React
- ใช้ Inertia.js หรือ API + React SPA

```
npm install react react-dom @inertiajs/inertia-react
```

Laravel + Livewire

- ไม่ต้องเขียน JS เอง
- เหมาะสำหรับ admin dashboard, form, modal
- ใช้ง่ายมาก เช่น:

```
@livewire('counter')
class Counter extends Component {
    public $count = 0;
    public function increment() { $this->count++; }
}
```

☐ **ตัวอย่าง Laravel Ecosystem**

Tool	คำอธิบาย
Laravel Forge	จัดการ VPS, deploy project, setup SSL โดยไม่ต้อง SSH
Laravel Vapor	Serverless PHP บน AWS (ใช้ Lambda, S3, RDS)
Laravel Nova	Admin Panel แบบ Premium (เสียเงิน)
Laravel Sail	Docker Compose สำหรับ local dev
Laravel Mix	Compile Assets (JS, CSS, Vue, React)
Laravel Sanctum	Token-based API Auth ง่ายๆ
Laravel Cashier	ระบบ Billing (Stripe, Paddle)
Laravel Scout	Full-text search (เช่น Algolia, Meilisearch)
Laravel Horizon	UI สำหรับ Queue Monitoring
Laravel Telescope	Debugging Tool เช่น Log, DB, Mail, Exception

---

☐ **แนวคิดที่ Laravel สนับสนุน**

☐ **Convention Over Configuration**

- ไม่ต้องกำหนดโครงสร้างซ้ำซ้อน
- ทำให้พัฒนาได้เร็ว

☐ **Clean Architecture / Hexagonal**

- Laravel รองรับ Layered Architecture ชัดเจน
- สนับสนุนการแยก Domain Logic, Application Service, Infrastructure

☐ **Test-Driven Development (TDD)**

- มี PHPUnit ในตัว
- รองรับ Feature, Unit, Integration Test

☐ **RESTful และ SPA/Microservices**

- ทำ API แบบ RESTful ง่ายมาก
- รองรับการทำ Frontend แยกจาก Backend หรือแยกเป็น Microservices

---

☐ **สรุปแนวโน้ม Laravel ในอีก 3–5 ปีข้างหน้า**

แนวโน้ม	รายละเอียด
☐ Monolith + Modular	Laravel สนับสนุนสถาปัตยกรรมแบบ Modular อย่างดี

แนวโน้ม	รายละเอียด
☐ Full-stack DX	Laravel + Livewire/Inertia.js ให้ประสบการณ์แบบ Full-stack
☐ DevOps	Sail, Forge, Vapor ทำให้ Laravel พร้อมใช้งาน production
☐ ความปลอดภัย	Laravel มี built-in CSRF, SQL Injection Protection, Auth
☐ Business Tools	Nova, Telescope, Horizon ช่วยให้ Laravel เป็น Enterprise Ready

---

การเปรียบเทียบ **Laravel (PHP)** กับ **JSF/Jakarta Faces (Java)** และ **Django (Python)** จำเป็นต้องพิจารณาหลายแง่มุม ได้แก่ สถาปัตยกรรม แนวคิด ความง่ายในการพัฒนา เครื่องมือ Ecosystem ความเหมาะสมของงาน และแนวโน้มในอนาคต โดยด้านล่างนี้คือการเทียบ **แบบละเอียด – จุดเด่น / จุดด้อย** และ **สถานการณ์ที่เหมาะสม**:

☐ **สรุปเปรียบเทียบภาพรวม**

คุณสมบัติหลัก	Laravel (PHP)	JSF / Jakarta Faces (Java)	Django (Python)
ภาษา	PHP	Java	Python
แนวคิดหลัก	MVC	Component-based MVC	MTV (Model–Template–View)
ความนิยม	สูงมาก (โดยเฉพาะ Web App, REST API)	ลดลง (ใช้เฉพาะองค์กร/รัฐ/Legacy Java)	สูงมากในสาย Data / AI / Web
Learning Curve	ง่าย (เริ่มไว)	ยาก (Enterprise-level)	ปานกลาง (Logic-first)
Developer Experience (DX)	ดีมาก (CLI, Artisan, Scaffolding)	ค่อนข้างต่ำ	ดี (Pythonic, Structured)
Speed to Develop	เร็ว	ช้า	เร็ว
ใช้ในงาน	Web App, CMS, e-Commerce, SaaS	ระบบภาครัฐ/รัฐวิสาหกิจ/องค์กรใหญ่	Data App, REST API, Web App, CMS
Community & Resources	ใหญ่มาก	จำกัด (แต่มีในโลก Java)	ใหญ่มาก โดยเฉพาะนักวิทย์ข้อมูล
Hosting	Shared Hosting ก็ได้	ต้องการ Java Server (Tomcat/Payara)	VPS, WSGI, Gunicorn, Docker

---

☐ **จุดเด่นของแต่ละ Framework**

☐ **Laravel (PHP) – จุดเด่น**

- เรียนรู้ง่าย ตอบโจทย์ dev มือใหม่-กลาง
- Full-featured (Auth, ORM, Validation, Mail, Queue, Testing)
- Ecosystem ครบ: Sail (Docker), Forge (Deploy), Nova (Admin), Vapor (Serverless)
- รองรับ REST API และ SPA (ผ่าน Vue/React/Livewire)
- Artisan CLI สร้าง scaffold ได้ไว
- Template Engine (Blade) ใช้ง่าย และ custom ได้
- มีความยืดหยุ่นสูง แต่ยังมีโครงสร้างที่ชัดเจน

☐ **JSF / Jakarta Faces (Java) – จุดเด่น**

- สร้าง UI แบบ **component-based** (คล้าย React ในแบบ server-side)
- มี integration กับ Java EE เต็มรูปแบบ (JPA, CDI, EJB, Bean Validation)
- ความปลอดภัยระดับ Enterprise
- ใช้ XML/Facelets สร้างหน้าเว็บ
- เหมาะสำหรับ **ระบบขนาดใหญ่**, ภาครัฐ หรือองค์กรที่มี Java infrastructure อยู่แล้ว

☐ **Django (Python) – จุดเด่น**

- "Batteries-included": มีครบทุกอย่างตั้งแต่เริ่ม
- Admin UI อัตโนมัติ (Django Admin)
- ORM ที่เข้าใจง่าย
- ใช้ Python ซึ่งเป็นภาษาที่อ่านง่าย มี community ใหญ่
- เหมาะกับ Web + Data + Machine Learning
- มี Testing tools และ REST Framework (DRF) ที่ดี

☐ **จุดด้อยของแต่ละ Framework**

☐ **Laravel (PHP) – จุดด้อย**

- PHP ยังมีภาพลักษณ์เก่าที่ยากจะลบ (แม้ Laravel จะดีมาก)
- Performance ด้อยกว่า Node.js หรือ Go หากโหลดสูงมาก
- ORM (Eloquent) ใช้ง่าย แต่ performance ไม่ดีใน query ซับซ้อน
- ต้องปรับตัวถ้าจะทำ Clean Architecture จริงจัง (ต้องใช้ Package เสริม เช่น Domain Layer)

☐ **JSF / Jakarta Faces – จุดด้อย**

- Learning curve สูง: Java + JSF XML config + server lifecycle
- พัฒนา UI ยุ่งยาก เทียบกับ frontend สมัยใหม่ (React/Vue)
- ใช้ทรัพยากรเครื่องสูง และช้าหากไม่ optimize
- Modern frontend integration (JS/SPA) ยาก
- หลายองค์กร migrate ออกไปยัง Spring Boot + REST + SPA แล้ว

☐ **Django (Python) – จุดด้อย**

- Template engine ค่อนข้างเรียบ ไม่ยืดหยุ่นเท่า Blade (Laravel)
- ORM มีข้อจำกัดใน query ซับซ้อนมาก (เทียบกับ SQLAlchemy ยังด้อย)
- ถ้าจะใช้กับ SPA ต้องแยก frontend อย่างชัดเจน (ไม่เหมือน Inertia ของ Laravel)
- ใช้งานยากขึ้นถ้าต้องจัดการ background task (Celery ต้อง config มาก)

☐ **ความเหมาะสมของงานแต่ละ Framework**

ประเภทโปรเจกต์	Laravel	JSF	Django
CMS / Blog / e-Commerce	☐ เหมาะมาก (มี Laravel Packages พร้อมใช้)	☐	☐
Web App แบบทั่วไป	☐	☐	☐
REST API	☐ (Sanctum, Passport)	☐ (ยาก, ไม่ออกแบบมาเพื่อ API)	☐ (Django REST Framework)
SPA/Frontend แยก	☐ (API + Vue/React/Livewire)	☐ (UI ฝั่ง Server ไม่ตอบโจทย์ SPA)	☐
ระบบองค์กร (HR, ERP, ภาครัฐ)	☐	☐	☐
งาน Big Data / AI / ML	☐	☐	☐ (Django ใช้ Python ซึ่งเหมาะ)
Real-time / WebSocket	⚠☐ (ต้องใช้ Laravel Echo, Pusher)	☐	⚠☐ (ต้องใช้ Channels เพิ่ม)

☐ **แนวโน้มในอนาคต**

Framework	แนวโน้ม
Laravel	เติบโตต่อเนื่องในกลุ่ม web dev ทั่วไป, SaaS, startup, และ DevOps (Docker, Serverless)
JSF	ลดลงเรื่อย ๆ โดยเฉพาะในกลุ่ม startup และ modern stack แต่ยังมีในภาครัฐ/องค์กร
Django	เติบโตในกลุ่ม Data Science, AI-driven web app, และ REST API สำหรับ Mobile/SPA

☐ **ข้อแนะนำจากผู้พัฒนา (เชิงกลยุทธ์)**

ถ้าคุณเป็น...	เลือกใช้

ถ้าคุณเป็น...	เลือกใช้
Dev มือใหม่ ⟶ กลาง	□ Laravel หรือ Django
ทีมพัฒนา Web App + SaaS	□ Laravel
บริษัทสายข้อมูล / AI	□ Django
ทีม Java Legacy หรือ หน่วยงานราชการ	□ JSF หรือ Spring
อยากทำ Fullstack ด้วยภาษาเดียว	□ JSF, □ Django ⟶ □ Laravel (PHP) หรือ Node.js (JS)

องค์กร บริษัท ห้างร้าน และหน่วยงานต่าง ๆ ที่ใช้ Laravel

องค์กร บริษัท ห้างร้าน และหน่วยงานต่าง ๆ ที่ใช้ Laravel โดยแบ่งตามหมวดหมู่ พร้อมตัวอย่างจริงจากทั้ง **ในประเทศ (ไทย)** และ **ต่างประเทศ** รวมถึง **แหล่งอ้างอิง** และ **ลักษณะการใช้งาน** ที่สอดคล้องกับ Laravel

---

□□ ตัวอย่างองค์กรในประเทศไทยที่ใช้ Laravel

1. หน่วยงานราชการ / มหาวิทยาลัย

หน่วยงาน	การใช้งาน Laravel
กรมทรัพย์สินทางปัญญา (DIP)	พัฒนา API เชื่อมโยงกับระบบภายใน (พบ Laravel Header ใน response)
สำนักงานพัฒนารัฐบาลดิจิทัล (DGA)	ใช้ Laravel บางส่วนในระบบ e-Service และ Backend ภายใน
มหาวิทยาลัยหลายแห่ง เช่น ม.บูรพา, ม.รังสิต	ใช้ Laravel ในระบบลงทะเบียน, ระบบวิจัย, ระบบสหกิจ
องค์การบริหารส่วนตำบล / เทศบาล	จ้าง Outsource เขียน Laravel CMS/Portal เช่น ข่าว, ลงทะเบียน, สำรวจ

□ Laravel ถูกใช้เพราะพัฒนาเร็ว, ราคาคุ้มค่ากับงบราชการ, และสามารถใช้กับ shared hosting ได้ง่าย

---

2. บริษัทเอกชนในไทย

บริษัท	การใช้งาน

บริษัท	การใช้งาน
Bitkub (Crypto Exchange)	ใช้ Laravel สำหรับ backend ระบบภายในและระบบจัดการ
Fastwork.co (Freelance Marketplace)	Laravel ใช้กับระบบ API ภายใน
SCB 10X	ทีม Startup ใช้ Laravel สำหรับ prototype และ MVP
บริษัทซอฟต์แวร์อย่าง Nextzy, Wazzadu, The Flight 19	Laravel ใช้ในโปรเจกต์ลูกค้าแบบ Enterprise / Startup
ร้านค้าออนไลน์ขนาดกลาง	ระบบ POS, Stock, E-commerce มักใช้ Laravel + Vue/Livewire

---

3. โรงเรียน/สถาบันการศึกษา

- Laravel นิยมใช้สร้างระบบ:
  - ลงทะเบียนนักเรียน
  - แบบฟอร์มออนไลน์
  - ระบบเก็บคะแนน / เอกสารวิชาการ
- เช่น โรงเรียนสาธิตจุฬา, รร.เซนต์คาเบรียล, รร.ในเขต กทม.หลายแห่ง

☐ **องค์กรและบริษัทต่างประเทศที่ใช้ Laravel**

1. Startup และบริษัทเทคโนโลยี

บริษัท	การใช้งาน Laravel
Laravel Forge, Vapor, Envoyer	พัฒนาโดยผู้สร้าง Laravel เอง
Invoice Ninja	ระบบบัญชีแบบ Open Source เขียนด้วย Laravel
October CMS	CMS ชื่อดัง พัฒนาโดยใช้ Laravel
Statamic (Flat-file CMS)	ใช้ Laravel เป็นฐาน
Flarum (Forum Software)	ใช้ Laravel สำหรับ backend API
AsgardCMS, PyroCMS	ทั้งคู่สร้างบน Laravel

---

2. องค์กรระดับโลก

องค์กร	รายละเอียด

องค์กร	รายละเอียด
BBC (UK)	ใช้ Laravel บางระบบสำหรับ internal tools
TourRadar (แพลตฟอร์มจองทัวร์)	ใช้ Laravel สำหรับ API และ Booking System
Crowdcube (UK)	Laravel ถูกใช้เป็น backend สำหรับบริการระดมทุน
AlphaCode (South Africa)	ใช้ Laravel สร้างแพลตฟอร์มเรียนรู้เขียนโค้ด
9GAG (บางระบบ)	Laravel ใช้ใน CMS ส่วนหลังบ้าน
Toyota (บางประเทศ)	ใช้ Laravel สร้าง dashboard ภายในแผนก HR และ CRM

☐ **ตัวอย่างเว็บจริงที่ใช้ Laravel** ***(จาก builtwith.com และ GitHub)***

เว็บไซต์	รายละเอียด
cachethq.io	Status Page System (เหมือนระบบแจ้ง server down)
laracasts.com	แพลตฟอร์มเรียน Laravel ชั้นนำ
myrank.co.in	แพลตฟอร์มการศึกษาอินเดีย ใช้ Laravel
neverbounce.com	Email Verification API Platform
worldwalking.org	Health Tracking Web ใช้ Laravel

☐ **เหตุผลที่องค์กรเลือก Laravel**

1. **Speed to Market**: Dev เร็ว เหมาะกับ MVP / Startup
2. **Security + Maintenance**: มี CSRF, SQL Injection Protection
3. **Scalability**: รองรับได้ทั้ง Monolith และ Microservices
4. **DX ดี**: Artisan CLI, Laravel Mix, Sail, Livewire ช่วยให้ Dev สนุก
5. **มี Ecosystem ครบ**: Forge, Vapor, Nova, Breeze, Jetstream
6. **รองรับ Frontend Stack สมัยใหม่**: Vue, React, Inertia.js
7. **Community ใหญ่ + Learning Resource เยอะ**: ทำให้หาคนได้ง่าย

☐ **สรุป**

ประเภทองค์กร	ใช้ Laravel ทำอะไร
ราชการ / เทศบาล	e-Service, ระบบแบบฟอร์ม, ข่าวสาร, จัดซื้อจัดจ้าง
มหาวิทยาลัย / โรงเรียน	ระบบทะเบียน, วิจัย, ประเมินผล, สหกิจศึกษา
บริษัทเอกชน	ระบบ ERP, POS, สต๊อก, พนักงาน, REST API

ประเภทองค์กร	ใช้ Laravel ทำอะไร
Startup	MVP, SaaS, Backend สำหรับ Mobile
ต่างประเทศ	CMS, Dashboard, Platform, API, DevOps Tools

Laravel คืออะไร, จุดเด่น และแนวคิด MVC

☐ Laravel คืออะไร?

Laravel คือ **PHP Framework แบบ Open Source** สำหรับพัฒนา **เว็บแอปพลิเคชัน (Web Applications)** ที่มีโครงสร้างชัดเจน สะอาด และยืดหยุ่น โดยพัฒนาโดย **Taylor Otwell** และเปิดตัวครั้งแรกในปี **2011**

Laravel สร้างขึ้นเพื่อแก้ปัญหาข้อจำกัดของ Framework PHP ยุคก่อน เช่น CodeIgniter หรือ Zend โดยเน้นไปที่:

- การพัฒนาแบบ MVC ที่ชัดเจน
- การเขียนโค้ดที่ “อ่านง่าย”
- การพัฒนาระบบที่ “เร็ว” และ “ปลอดภัย”

☐ จุดเด่นของ Laravel

จุดเด่น	รายละเอียด
1. Developer Experience (DX) ดีเยี่ยม	Laravel มี Artisan CLI, Template Engine (Blade), ระบบ Routing, Migration, Testing ที่ใช้งานง่าย
2. มีทุกอย่างพร้อม (Full-Stack)	Authentication, Authorization, Mail, Queue, Notification, REST API, File Storage, Broadcasting
3. Ecosystem ครบ	Laravel Forge, Vapor, Nova, Breeze, Jetstream, Sail
4. ใช้งานร่วมกับ Frontend สมัยใหม่ได้ดี	รองรับ Vue, React, Livewire, Inertia.js
5. เรียนรู้ง่ายกว่า PHP Framework อื่น ๆ	มีโครงสร้างชัดเจน, Document ครบ, Community ใหญ่
6. ความปลอดภัยในตัว	CSRF Protection, XSS Protection, SQL Injection Safe
7. รองรับการขยายขนาด (Scalable)	ทั้งแบบ Monolith และ Microservices