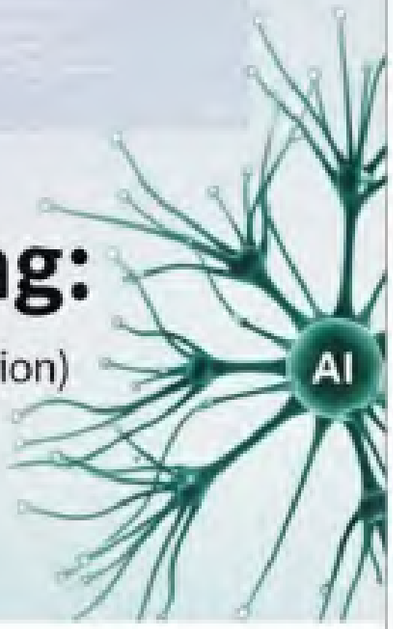
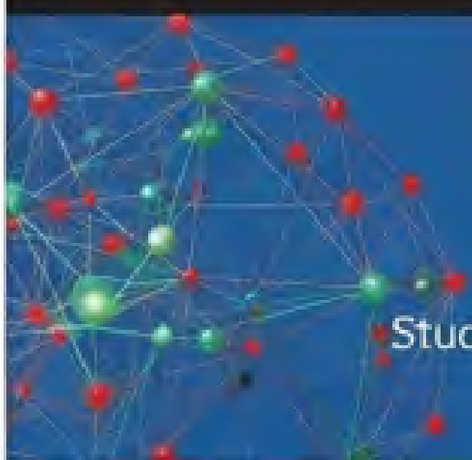
DaisyUI Web Programming:

Beginner (Integrative-Generative AI Edition)



Contents:

Author:



Student Price Book Center

คำนำ

DaisyUI Web Programming: Beginner

ในยุคที่การพัฒนาเว็บแอปพลิเคชันและเว็บไซต์ตอบสนองผู้ใช้อย่างรวดเร็ว การออกแบบ UI ที่สวยงาม มีมาตรฐาน และใช้งานง่ายกลายเป็นหัวใจสำคัญของความสำเร็จของโปรเจกต์ทุกประเภท **DaisyUI** ถือเป็นหนึ่งในเครื่องมือที่ตอบโจทย์ความต้องการนี้ได้อย่างชัดเจน โดยเป็น **Plugin ของ Tailwind CSS** ที่เข้ามาเติมเต็มจุดอ่อนของ Tailwind แบบเพียว ๆ ที่แม้จะมี Utility Class ที่ยืดหยุ่นสูง แต่กลับขาด Component สำเร็จรูปที่พร้อมใช้งาน การรวมคุณสมบัติของ Tailwind กับ Component สำเร็จรูปของ DaisyUI ช่วยให้นักพัฒนาสามารถสร้าง UI ได้รวดเร็ว ลดความซ้ำซ้อนของโค้ด และรักษามาตรฐานการออกแบบในเวลาเดียวกัน

หนังสือเล่มนี้ถูกออกแบบสำหรับผู้เริ่มต้นและนักพัฒนาที่ต้องการ **เรียนรู้การใช้งาน DaisyUI อย่างเป็นระบบ** ตั้งแต่พื้นฐานจนถึงการสร้าง UI ที่ซับซ้อน โดยเริ่มจากบทที่ 1 ซึ่งจะทำให้ผู้อ่านเข้าใจว่า DaisyUI คืออะไร มีประวัติความเป็นมา วิวัฒนาการ การใช้งานปัจจุบัน รวมถึงแนวโน้มในอนาคต การเปรียบเทียบ Tailwind เพียว ๆ กับ DaisyUI พร้อมขั้นตอนการติดตั้งผ่าน npm, yarn, หรือ pnpm และการกำหนดค่า DaisyUI Plugin ในไฟล์ tailwind.config.js นอกจากนี้ยังมีตัวอย่างบูรณาการเพื่อให้ผู้อ่านเห็นภาพการใช้งานจริง

ต่อเนื่องด้วยบทที่ 2: **DaisyUI Basics** ผู้อ่านจะได้เรียนรู้ **โครงสร้าง class สำเร็จรูป** การใช้ Layout Components พื้นฐาน เช่น navbar, footer, hero รวมถึงวิธีปรับแต่ง Components เหล่านี้ด้วย Tailwind Utility Class เพื่อสร้างหน้าเว็บที่สวยงามและตอบโจทย์การใช้งาน อีกทั้งบทนี้ยังมีตัวอย่างบูรณาการและ Ultimate Integrated Examples ที่ช่วยให้ผู้เรียนสามารถต่อยอดได้ทันที

บทที่ 3: **Forms & Inputs** จะเจาะลึกองค์ประกอบฟอร์มของ DaisyUI ตั้งแต่ปุ่ม (Button) และการเลือกสี Input, Select, Checkbox, Toggle รวมถึง Validation States เช่น success และ error ผ่านตัวอย่างบูรณาการและ Ultimate Integrated Examples ทำให้ผู้อ่านสามารถสร้างฟอร์มที่ใช้งานง่าย สวยงาม และเป็นมิตรกับผู้ใช้

บทที่ 4: **Theming เบื้องต้น** จะสอนวิธีการจัดการธีมใน DaisyUI ตั้งแต่ค่า Default Theme การใช้ data-theme เพื่อสลับธีม เช่น light, dark, cupcake และการปรับแต่งธีมตามความต้องการ พร้อมตัวอย่างบูรณาการและ Ultimate Example เพื่อให้ผู้เรียนเข้าใจการปรับธีมและโทนสีของเว็บอย่างลึกซึ้ง

บทที่ 5: **Components Overview** จะพาผู้อ่านไปรู้จัก Components สำคัญ เช่น card, modal, alert, badge, tabs และ steps พร้อมแนวคิดการรวมหลาย Component ให้สร้าง UI ที่ซับซ้อน เช่นหน้า Login Page โดยตัวอย่างบูรณาการและ Ultimate Example จะช่วยให้ผู้อ่านเข้าใจการใช้งาน Components แบบครบวงจร

หนังสือเล่มนี้จึงเป็นคู่มือที่ **ครบถ้วนและใช้งานได้จริง** สำหรับผู้เริ่มต้นที่ต้องการเรียนรู้ DaisyUI ตั้งแต่พื้นฐานจนถึงการสร้างเว็บที่ใช้งานได้จริง ด้วยโครงสร้างที่เรียงลำดับอย่างเป็นระบบและตัวอย่างบูรณาการตลอดเล่ม ผู้อ่านจะสามารถสร้างเว็บที่สวยงาม มีมาตรฐาน และปรับแต่งได้อย่างยืดหยุ่น พร้อมรับมือกับความต้องการของผู้ใช้ยุคใหม่

ด้วยรักและปรารถนาดี

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1

Introduction to DaisyUI

(Introduction to DaisyUI)

เนื้อหา

- Introduction to DaisyUI
- รายละเอียดเชิงลึก บทที่ 2 (DaisyUI Basics)
- ประวัติความเป็นมา, วิวัฒนาการ, สถิติการใช้งานปัจจุบัน และแนวโน้มในอนาคตของ daisyUI
- DaisyUI คืออะไร?
- เปรียบเทียบ Tailwind เพียว ๆ (Pure Tailwind) กับ DaisyUI
- การติดตั้ง DaisyUI (npm, yarn, pnpm)
- DaisyUI plugins ใน tailwind.config.js
- ตัวอย่างบูรณาการ

บทนำ บทที่ 1: Introduction to DaisyUI

DaisyUI ถือเป็นหนึ่งใน **Tailwind CSS Plugin ที่ได้รับความนิยมมากที่สุด** เนื่องจากมันเข้ามาแก้ไขปัญหาสำคัญของการพัฒนา UI ด้วย Tailwind แบบเพียว ๆ ซึ่งปัญหาหลักคือ Tailwind เองแม้จะเป็น Utility-First Framework ที่ยืดหยุ่นสูง แต่กลับ **ไม่มี Component ที่พร้อมใช้งาน** ทำให้นักพัฒนาต้องเสียเวลาเขียนโค้ดซ้ำซ้อนในการสร้างปุ่ม ฟอร์ม หรือ Layout พื้นฐาน DaisyUI จึงเข้ามาเติมเต็มส่วนนี้ โดยเพิ่มชุดของ Components ที่สามารถใช้งานได้ทันทีโดยไม่เสียความยืดหยุ่นของ Tailwind

หากเปรียบเทียบ **Tailwind เพียว ๆ** กับ **DaisyUI** จะเห็นภาพความแตกต่างได้อย่างชัดเจน Tailwind เพียว ๆ จะมอบเครื่องมือในการจัดการ CSS แบบ Utility Class ที่ละเอียดมาก แต่การสร้าง Component เช่น Button หรือ Card นักพัฒนาต้องประกอบเองทีละ Class ทำให้โค้ดมีความซ้ำซ้อนและอาจยากต่อการบำรุงรักษา ขณะที่ DaisyUI มาพร้อมกับ Component ที่ถูกออกแบบมาแล้ว เช่น btn, card, navbar ซึ่งสามารถเรียกใช้งานได้ทันที ลดเวลาและจำนวนบรรทัดโค้ดอย่างมาก

อีกหนึ่งข้อได้เปรียบที่ทำให้ DaisyUI ได้รับความนิยมคือ **ความง่ายในการติดตั้งและใช้งาน** ไม่ว่าคุณจะใช้ตัวจัดการแพ็กเกจแบบ npm, yarn, หรือ pnpm คุณสามารถติดตั้ง DaisyUI ได้ด้วยคำสั่งเพียงบรรทัดเดียว เช่น npm install daisyui จากนั้นเพียงเพิ่ม DaisyUI เข้าไปในไฟล์ tailwind.config.js คุณก็พร้อมใช้งาน Component ของ DaisyUI ได้ทันที

กระบวนการติดตั้ง DaisyUI ไม่เพียงแต่รวดเร็ว แต่ยังถูกออกแบบมาให้ **ทำงานร่วมกับ Tailwind ได้อย่างราบรื่น** เพราะ DaisyUI เป็นเพียง Plugin ที่ต่อยอดจาก Tailwind โดยตรง คุณยังคงสามารถใช้ Utility Class ของ Tailwind ได้ตามปกติ และเลือกใช้ Component ของ DaisyUI เพื่อเสริมประสิทธิภาพการพัฒนา UI ในจุดที่คุณต้องการ

หัวใจสำคัญของ DaisyUI อยู่ที่ **การทำงานผ่าน DaisyUI Plugin** ที่ถูกกำหนดในไฟล์ tailwind.config.js โดยเมื่อคุณเพิ่ม DaisyUI ลงในส่วนของ plugins ตัว Tailwind จะโหลด Component และ Theme ที่ DaisyUI เตรียมมาให้ทันที นักพัฒนาสามารถกำหนด Theme เอง หรือเลือกใช้ Theme ที่มีมาให้แล้ว เช่น light, dark, cupcake, bumblebee ทำให้โปรเจกต์ของคุณมีความยืดหยุ่นและตอบโจทย์การออกแบบที่หลากหลาย

การทำงานของ DaisyUI Plugin ใน tailwind.config.js ยังเปิดโอกาสให้คุณ **ปรับแต่งตามความต้องการเฉพาะ** ได้ เช่น การปิดบาง Component ที่ไม่จำเป็นต่อโปรเจกต์ การกำหนด Theme เฉพาะแบรนด์ หรือแม้กระทั่งการสร้าง Theme ใหม่ตั้งแต่ต้น สิ่งเหล่านี้ทำให้นักพัฒนาสามารถใช้ DaisyUI ได้ทั้งแบบรวดเร็วและแบบปรับแต่งได้อย่างลึกซึ้ง

โดยสรุปแล้ว บทที่ 1: Introduction to DaisyUI นี้จะทำให้ผู้อ่านเข้าใจ **ว่าทำไม DaisyUI ถึงสำคัญต่อการพัฒนา UI ด้วย Tailwind** ทั้งในมิติการแก้ปัญหา การเปรียบเทียบข้อดีเมื่อเทียบกับ Tailwind เพียว ๆ ตลอดจนขั้นตอนการติดตั้งและการกำหนดค่าใน tailwind.config.js ซึ่งทั้งหมดนี้คือพื้นฐานที่จำเป็นสำหรับการเริ่มต้นใช้งาน DaisyUI อย่างมีประสิทธิภาพในโปรเจกต์จริง

Introduction to DaisyUI

- DaisyUI คืออะไร และแก้ปัญหาอะไรของ Tailwind
- ความแตกต่างระหว่าง Tailwind เพียว ๆ vs DaisyUI
- การติดตั้ง DaisyUI (npm, yarn, pnpm)
- DaisyUI plugins ใน tailwind.config.js

---

1) DaisyUI คืออะไร และแก้ปัญหาอะไรของ Tailwind

- **ภาพรวมสั้น ๆ:**

  DaisyUI เป็น **plugin** สำหรับ Tailwind CSS ที่เพิ่มชุด *UI components* และ *theme system* แบบสำเร็จรูป (buttons, cards, modal, navbar, forms ฯลฯ) โดยใช้ประโยชน์จากคลาส utility ของ Tailwind — กล่าวคือเป็น CSS-only component library ที่ทำงานเป็น plugin บน Tailwind. (daisyui.com)

- **ปัญหาที่แก้ให้กับ Tailwind:**

1. **ลดงานออกแบบ/boilerplate** — แทนที่จะต้องประกอบชุดยูทิลิตี้เองเป็นปุ่ม/การ์ด/โมดัล ทีละชิ้น คุณสามารถใช้ class เดียวเช่น btn btn-primary เพื่อได้ UI พร้อมใช้ทันที
2. **ธีมและสีที่จัดการได้ง่าย** — มีระบบธีมสำเร็จรูป (built-in themes) และสามารถสร้างธีมของตัวเองได้ ทำให้เปลี่ยนลุคทั้งเว็บด้วยตัวแปรธีมเดียวได้เร็วขึ้น. (daisyui.com)
3. **ยังคงสอดคล้องกับ Tailwind** — เนื่องจาก DaisyUI สร้างบน Tailwind utilities คุณยังสามารถผสม/override ด้วย Tailwind ปกติได้ (ไม่ต้องเรียน CSS เพิ่มเยอะ) (codeparrot.ai)

---

2) ความแตกต่างระหว่าง Tailwind เพียว ๆ vs DaisyUI

- **Tailwind เพียว ๆ**
  - เป็น *utility-first CSS framework* — ให้คลาสเล็ก ๆ (เช่น px-4, text-sm, flex) เพื่อประกอบ UI เองทั้งหมด
  - ข้อดี: ยืดหยุ่นสูง, ขนาดไฟล์เล็ก (ถ้าพูดถูกต้อง purge)
  - ข้อจำกัด: ต้องออกแบบ component เองทั้งหมด ⟶ ฝั่ง UI ต้องใช้เวลาสร้าง pattern ซ้ำ ๆ
- **DaisyUI (ขยายจาก Tailwind)**
  - ให้ *component-level classes* (opinionated) เช่น btn, card, modal ที่ห่อหลายๆ utility ไว้แล้ว ⟶ เร็วกว่าการประกอบจาก utility ทีละตัว
  - มี **ธีมสำเร็จรูป** (หลายสิบธีม) และระบบ custom theme ที่ใช้ CSS variables ⟶ เปลี่ยนสี/โทนทั้งโปรเจกต์ได้ง่ายกว่า
  - ข้อแลกเปลี่ยน: ถ้าต้องการ custom ละเอียดมาก ๆ บางครั้งต้อง override/extend เพิ่ม แต่โดยทั่วไปช่วยเร่งพัฒนาได้มาก (ดีสำหรับ prototyping, dashboards, SaaS) (daisyui.com)

---

3) การติดตั้ง DaisyUI (npm, yarn, pnpm, Bun)

- **คำสั่งติดตั้ง (ตัวอย่าง)** — คำสั่งตามเอกสารอย่างเป็นทางการ:
  - npm: npm i -D daisyui@latest
  - pnpm: pnpm add -D daisyui@latest
  - yarn: yarn add -D daisyui@latest
  - bun: bun add -D daisyui@latest
    (นอกจากนี้ docs ยังแนะนำวิธีติดตั้งพร้อม Tailwind สำหรับ Next.js/CRA ฯลฯ). (daisyui.com)
- **การเชื่อมต่อกับไฟล์ CSS (สองแนวทางหลัก)**

1. **แบบดั้งเดิม — Require ใน tailwind.config.js**
    - ติดตั้งแล้วเพิ่มใน plugins:
2. // tailwind.config.js
3. module.exports = {
4. content: ["./src/**/*.{html,js,jsx,ts,tsx}"],
5. theme: { extend: {} },
6. plugins: [
7. require('daisyui'),
8. // ...plugins อื่น ๆ เช่น @tailwindcss/forms
9.],
10. }

    วิธีนี้เป็นวิธีที่ถูกใช้งานบ่อยและรองรับ Tailwind รุ่นก่อนหน้า/ทั่วไป. (Tailkits)
11. **(ใหม่ใน v5) — ใช้ @plugin directive ในไฟล์ CSS**
    - คุณสามารถเรียก plugin จากไฟล์ CSS/entry ของคุณ ด้วย @plugin "daisyui"; (และสามารถให้ config ภายในบล็อก { ... }) เช่น:
12. /* app.css (หรือ globals.css) */
13. @tailwind base;
14. @tailwind components;
15. @tailwind utilities;
16.
17. @import "tailwindcss";
18. @plugin "daisyui";

    หรือใส่ config ข้างในบล็อก (ตัวอย่างด้านล่างหัวข้อถัดไป). (daisyui.com)

- **ตัวอย่างการติดตั้งแบบ Next.js (สรุปขั้นตอน)**
    0. สร้างโปรเจกต์ Next.js
    1. ติดตั้ง tailwindcss, postcss, autoprefixer, daisyui
    2. เพิ่ม daisyui เป็น plugin (แบบ require('daisyui') หรือ @plugin) และ import CSS ใน _app/globals

    (เอกสารมีตัวอย่าง step-by-step สำหรับ Next.js). (daisyui.com)

---

4) DaisyUI plugins ใน tailwind.config.js (การตั้งค่า + ตัวเลือกสำคัญ)

มี 2 แนวทางที่เราควรเข้าใจ: (A) ใส่ require('daisyui') ใน tailwind.config.js และกำหนดตัวเลือก daisyui ที่นั่น — หรือ (B) ใช้ @plugin "daisyui" พร้อมบล็อก config ในไฟล์ CSS (v5)

A — ตัวอย่างตั้งค่าใน tailwind.config.js

```
// tailwind.config.js
module.exports = {
  content: ["./src/**/*.{html,js,jsx,ts,tsx}"],
  theme: { extend: {} },
  plugins: [
    require('daisyui'),
  ],
  // คอนฟิกสำหรับ daisyui (ถ้าใช้แบบ require())
  daisyui: {
    themes: ["light", "dark", "cupcake"], // หรือ object สำหรับ custom theme
    logs: true,   // เปิด/ปิด log ของ daisyUI
    // มี options เพิ่ม เช่น prefix, root, include, exclude (v5 เพิ่มความสามารถ)
  }
}
```

- themes สามารถเป็น array ของชื่อธีม built-in หรือ object ที่นิยามสีของคุณเองได้ (custom theme). (daisyui.com)

B — ตัวอย่างใช้ @plugin + บล็อก config (v5)

```
/* app.css */
@tailwind base;
@tailwind components;
@tailwind utilities;

@import "tailwindcss";
@plugin "daisyui" {
  themes: light --default, dark --prefersdark;
  root: ":root";
  include: ;
  exclude: ;
  prefix: ;
  logs: true;
}
```

- รูปแบบ @plugin "daisyui" { ... } จะใช้สำหรับ daisyUI v5 ขึ้นไป ซึ่งเพิ่มความสามารถเช่น include/exclude เพื่อเลือกโหลดเฉพาะบาง component (ลดขนาด CSS ถ้าต้องการ) — ฟีเจอร์นี้ทำให้เลือก **include เฉพาะส่วน** (เช่น toggle) หรือ **exclude บางส่วน** ได้. (daisyui.com)

คำอธิบาย option ที่ควรรู้ (สรุป)

- themes — กำหนดธีมที่ต้องการให้มี (ชื่อ built-in หรือ custom object). (daisyui.com)
- root — กำหนด selector สำหรับตัวแปร CSS root (ค่า default เป็น :root). (daisyui.com)
- include / exclude — (v5) ระบุเฉพาะ component ที่ต้องการหรือไม่ต้องการให้ถูกสร้างเป็น CSS เพื่อประหยัดขนาดไฟล์. (daisyui.com)
- prefix — ถ้าต้องการหลีกเลี่ยงชื่อ class conflict กับไลบรารีอื่น ๆ สามารถตั้ง prefix ให้กับ class ของ daisyUI ได้ (เช่น daisy-). (daisyui.com)
- logs — ควบคุมการแสดง log ของ daisyUI ขณะ build/dev. (daisyui.com)

Tips / Gotchas & Best Practices (สั้น ๆ)

- หากคุณใช้ **Tailwind v4** ให้ตรวจสอบ compatibility — daisyUI v5 แนะนำให้ใช้งานร่วมกับแนวทาง @plugin ของ Tailwind v4+ (ดูเอกสาร upgrade guide ก่อนอัพเกรด). (daisyui.com)
- ถ้าต้องการ **ขนาด CSS เล็กลง** ให้ใช้ include/exclude (v5) เพื่อสร้างเฉพาะ component ที่ต้องการเท่านั้น. (daisyui.com)
- ถ้าต้องการ **ธีมที่เกินจาก built-in** ให้สร้าง custom theme ใน daisyui.themes (หรือผ่าน CSS variables) — docs และ theme generator ช่วยให้สร้างค่าที่สอดคล้อง (OKLCH generator). (daisyui.com)
- หากใช้ **CDN** (ไม่ build ด้วย Tailwind) — การ custom theme อาจซับซ้อนกว่า (ดู discussion ใน repo) — แนะนำใช้แบบติดตั้งผ่าน npm/pnpm/yarn สำหรับการ customize ที่ลึก. (GitHub)

สรุปสั้น ๆ (สิ่งที่คุณได้จากบทนี้)

- DaisyUI = **Tailwind plugin** ที่ให้ component พร้อมใช้ + theme system ⟶ เร็วขึ้นในการสร้าง UI. (daisyui.com)
- ติดตั้งง่าย (npm/pnpm/yarn/bun) แล้วเพิ่มเป็น plugin (ผ่าน tailwind.config.js หรือ @plugin ในไฟล์ CSS ขึ้นกับเวอร์ชัน). (daisyui.com)
- ตั้งค่า (themes, prefix, include/exclude, logs ฯลฯ) ได้ทั้งใน tailwind.config.js หรือในบล็อก @plugin (v5) — v5 เพิ่มความสามารถให้ปรับได้ละเอียดขึ้นเพื่อประสิทธิภาพ. (daisyui.com)

รายละเอียดเชิงลึก บทที่ 2 (DaisyUI Basics)

สรุปสั้น ๆ (ภาพรวมเชิงเทคนิค)

- **DaisyUI = Tailwind plugin** ที่ให้ชุด *component class* (เช่น btn, card, modal…) และระบบธีมโดยใช้ **CSS variables** (semantic color tokens เช่น primary, secondary, base-100, base-content ฯลฯ) แทนการเขียน utility ครบทุกตัวเอง ช่วยเร่งพัฒนา UI แบบ

Tailwind-first โดยยังเป็น *pure CSS* (ไม่มี runtime JS จำเป็น) ซึ่งเวอร์ชัน 5 ย่อแพ็กเกจและไฟล์ CSS ลง พร้อมเพิ่มความสามารถด้าน include/exclude และการ config ในไฟล์ CSS. (daisyui.com)

---

1) สถาปัตยกรรม + หลักการทำงาน (how it works)

- DaisyUI ทำงานเป็น **Tailwind plugin** — ในหลังบ้านมันสร้างชุดคลาส component โดยผสม/ใช้ utility rules ของ Tailwind และกำหนดค่าธีมผ่าน **CSS variables** (เช่น --color-primary, --color-primary-content, --color-base-100 ฯลฯ) — เมื่อสลับธีม (ผ่าน data-theme หรือ input-based theme controller) มันแค่เปลี่ยนค่าตัวแปรเท่านั้น ทำให้การสลับธีมเป็นเรื่องเบาและทันที. (daisyui.com)
- ข้อสังเกตสำคัญ: **ไม่มี JS ตายตัวสำหรับคอมโพเนนท์** (คุณยังสามารถใช้ pattern แบบ HTML-only เช่น checkbox-toggle หรือ <dialog> หรือจัดการด้วย JS/React ตามต้องการ) — นั่นแปลว่า DaisyUI ลด bundle size แต่ให้ความยืดหยุ่นในการผสานกับ framework ที่ต่างกัน. (daisyui.com)

---

2) ระบบธีม & CSS variables (เชิงลึก)

Semantic tokens ที่ DaisyUI ใช้ (ตัวอย่าง)

DaisyUI ให้ชุดชื่อสีแบบ semantic ที่ใช้ทั่วทั้งไลบรารี — แต่ละตัวจะมีตัวแปร CSS ที่สอดคล้อง (ตัวอย่างชื่อ-ตัวแปร):

- primary ⟶ --color-primary
- primary-content ⟶ --color-primary-content
- secondary ⟶ --color-secondary
- accent ⟶ --color-accent
- neutral ⟶ --color-neutral
- base-100, base-200, base-300, base-content ⟶ --color-base-100 ฯลฯ
- info, success, warning, error และคู่ -content ของแต่ละตัว ⟶ --color-info, --color-info-content ฯลฯ. (daisyui.com)

(เพราะใช้ตัวแปรแบบ --color-* ทำให้คุณสามารถใช้ค่าที่เป็น oklch() หรือ hsl() เพื่อได้ control ด้าน accessibility และ opacity แบบ native — v5 สนับสนุนรูปแบบสมัยใหม่ เช่น oklch().) (LogRocket Blog)

ทำไมใช้ semantic variables ดี

- ใช้คลาสเช่น bg-primary, text-primary-content แทนการพึ่งเฉพาะ bg-blue-500 ⟶ ทำให้เปลี่ยนโทนสีทั้งโปรเจกต์ได้ด้วยการเปลี่ยนตัวแปรธีมเดียว

- รองรับการปรับ opacity แบบ Tailwind (/60 notation) เพราะค่าถูกเก็บในตัวแปรสีที่ Tailwind/modern CSS รองรับ. (daisyui.com)

---

3) วิธีตั้งค่า (install + config options) — ตัวอย่างจริง

ติดตั้ง

```
# npm
npm install -D daisyui@latest

# pnpm
pnpm add -D daisyui@latest

# yarn
yarn add -D daisyui@latest
```

(เวอร์ชัน v5 ต้องจับคู่กับ Tailwind เวอร์ชันที่รองรับ — ตรวจสอบ compatibility ก่อนอัพเกรด). (daisyui.com)

วิธีเชื่อม (สองแนวทางสำคัญ — ขึ้นกับ Tailwind/DaisyUI เวอร์ชัน)

A) เก่า/ทั่วไป — tailwind.config.js + require('daisyui')

```
// tailwind.config.js
module.exports = {
  content: ["./src/**/*.{html,js,jsx,ts,tsx}", "node_modules/daisyui/dist/**/*.js"],
  theme: { extend: {} },
  plugins: [ require('daisyui') ],
  daisyui: {
    themes: ["light", "dark"], // หรือ custom object
  }
}
```

หมายเหตุ: ถ้าคุณใช้ไลบรารี UI ที่แปะ class แบบ runtime (เช่น react-daisyui) ให้เพิ่ม node_modules/... เข้า content เพื่อไม่ให้ Tailwind purge คลาสเหล่านั้นออก. (GitHub)

B) (แนะนำใน v5) — ใส่ config ในไฟล์ CSS ด้วย @plugin "daisyui" (CSS-first)

```
/* app.css (หรือ globals.css) */
@tailwind base;
@tailwind components;
@tailwind utilities;
```

```
@import "tailwindcss";

/* DaisyUI plugin + config (v5) */
@plugin "daisyui" {
  themes: light --default, dark --prefersdark;
  root: ":root";
  include: ;
  exclude: ;
  prefix: ;
  logs: true;
}
```

- ข้อดี: v5 อนุญาตให้ config ใกล้กับจุดที่คุณ import Tailwind ทำให้ workflow แบบ CSS-first สะดวก และ v5 เพิ่มความสามารถ include/exclude เพื่อเลือกเฉพาะชิ้นส่วนของไลบรารีที่ต้องการ (ลดขนาด CSS). (daisyui.com)

---

4) ตัวอย่างการสร้าง Custom Theme (3 แบบให้เลือก)

ตัวอย่าง A — custom theme ใน tailwind.config.js (v4/v4-style)

```
// tailwind.config.js
module.exports = {
  // ...content, theme, plugins
  daisyui: {
    themes: [
      {
        mytheme: {
          "primary": "#06b6d4",
          "primary-focus": "#0891b2",
          "primary-content": "#ffffff",
          "secondary": "#f97316",
          "accent": "#8b5cf6",
          "neutral": "#111827",
          "base-100": "#ffffff",
          "base-200": "#f3f4f6",
          "base-300": "#e5e7eb",
          "base-content": "#111827",
```

```
        "info": "#0284c7",
        "success": "#16a34a",
        "warning": "#f59e0b",
        "error": "#dc2626"
      }
    }
  ]
 }
}
```

(วิธีนี้ยังใช้ได้ แต่ v5 แนะนำไฟล์ CSS approach). (Stack Overflow)

ตัวอย่าง B — custom theme (v5) โดยใช้ CSS variables / OKLCH (จากตัวอย่างสมัยใหม่)

```
@import "tailwindcss";
@plugin "daisyui";

/* define theme with OKLCH (cleaner color manipulation) */
@plugin "daisyui/theme" {
  name: "mytheme";
  default: true;
  prefersdark: false;
  color-scheme: "light";
  --color-primary: oklch(84.19% 0.16 88.33);
  --color-primary-content: oklch(12% 0.01 20);
  --color-secondary: oklch(62% 0.10 300);
  --color-base-100: oklch(98% 0.005 220);
  --color-base-content: oklch(10% 0.02 240);
  /* ...other variables... */
}
```

(ตัวอย่างรูปแบบการกำหนดสีด้วย oklch() อ้างอิงจากตัวอย่างการใช้งานในบทความแนะนำ daisyUI v5). (LogRocket Blog)

ตัวอย่าง C — ปรับแค่ตัวแปร CSS หลังจาก @plugin (quick override)

```
@plugin "daisyui" {
  themes: light --default, dark --prefersdark;
}
```

```
/* แล้ว override บางตัวแปร */
:root {
  --color-primary: #0ea5a4;
  --color-primary-content: #ffffff;
}
```

(สะดวกเมื่อต้องการ patch สีบางตัวโดยไม่ต้องนิยามธีมทั้งชุด). (daisyui.com)

---

5) Include / Exclude / Prefix (ลดขนาด / หลีกเลี่ยงการชนชื่อ)

- **Include / Exclude:** v5 อนุญาต include เพื่อโหลดเฉพาะ component ที่ต้องการ (เช่น include: toggle;) หรือ exclude เพื่อไม่โหลดบางส่วน (เช่น exclude: scrollbar;) — เหมาะกับโปรเจกต์ที่อยากลด CSS footprint หรือ CDN micro-css usage. (daisyui.com)
- **Prefix:** ถ้ามีความเสี่ยงชื่อ class ชนกับไลบรารีอื่น สามารถใส่ prefix เพื่อให้ทุก class ของ daisyUI ถูก prefix (เช่น daisy-btn) — ช่วย adopt แบบ incremental บนโปรเจกต์เดิม. (มี discussion/issue เกี่ยวกับเรื่องนี้ใน repo). (daisyui.com)

---

6) การใช้ DaisyUI กับ React — ตัวอย่างปฏิบัติ (modal + theme persistence)

A) Theme switch + persist (React)

```
// ThemeSwitcher.jsx (simple)
import { useEffect, useState } from "react";

export default function ThemeSwitcher(){
  const [theme, setTheme] = useState(() => localStorage.getItem("theme") || "light");

  useEffect(() => {
    document.documentElement.setAttribute("data-theme", theme);
    localStorage.setItem("theme", theme);
  }, [theme]);

  return (
    <div className="flex gap-2 items-center">
      <button className="btn" onClick={() => setTheme("light")}>Light</button>
      <button className="btn btn-primary" onClick={() =>
setTheme("mytheme")}>MyTheme</button>
      <button className="btn btn-ghost" onClick={() => setTheme("dark")}>Dark</button>
```

```
    </div>
  );
}
```

- วิธีนี้สอดคล้องกับ Theme Controller ตัวอย่างใน docs และสามารถเก็บค่าใน localStorage เพื่อความถาวร. (daisyui.com)

B) Modal — แนะนำ 3 วิธี (docs แนะนำ <dialog> เป็นทางเลือกที่ accessible)

- **วิธีแนะนำ (accessible):** ใช้ <dialog> + showModal() / close() — ควบคุมด้วย JS/React (keyboard accessible, Esc ปิดได้).
- **HTML-only:** Checkbox toggle pattern (.modal-toggle) — ดีสำหรับ HTML-only flows แต่ถ้าใช้ React ควร bind checked/onChange ให้ถูกต้อง.
- **URL-based:** เปิดตามพารามิเตอร์ URL — เหมาะกับบาง use-case. (daisyui.com)

ตัวอย่าง <dialog> ใน React:

```
import { useRef } from "react";

export default function DialogExample(){
  const dialogRef = useRef(null);
  return (
    <>
      <button className="btn" onClick={() => dialogRef.current?.showModal()}>Open
Dialog</button>

      <dialog ref={dialogRef} className="modal">
        <form method="dialog" className="modal-box">
          <h3 className="font-bold text-lg">Confirm</h3>
          <p className="py-4">Do you confirm this action?</p>
          <div className="modal-action">
            <button className="btn" onClick={() => dialogRef.current?.close()}>Close</button>
          </div>
        </form>
      </dialog>
    </>
  );
}
```

(ถ้าเปลี่ยนไปใช้ checkbox-controlled modal ให้ใช้ checked={isOpen} + onChange ใน React เพื่อรักษา state สอดคล้องกับ React model). (daisyui.com)

---

7) Performance & Production notes

- **v5 ลดขนาดมาก** — รายงานว่า package/ไฟล์ CDN เล็กลงอย่างมีนัยสำคัญ (ตัวเลขระบุใน release notes) และ v5 เพิ่ม micro-css + include/exclude เพื่อโหลดเฉพาะสิ่งที่ต้องการ. ถ้าต้องการขนาด CSS เล็กสุด ให้ใช้ include เพื่อเลือกเฉพาะ component ที่ใช้ หรือใช้ CDN micro files สำหรับแต่ละ component. (daisyui.com)
- **Purge / Content scanning:** หากคุณใช้ไลบรารีที่สร้าง class แบบ runtime (เช่น react-daisyui) หรือสร้าง class ชื่อโดย string-concat ใน JS ให้แน่ใจว่า tailwind.config.js ของคุณมี content ที่ครอบคลุม (รวม node_modules/daisyui/dist/**/*.js หรือ node_modules/react-daisyui/dist/**/*.js) หรือใช้ safelist เพื่อไม่ให้ Tailwind purge คลาสที่ต้องการ. (GitHub)
- **Version compatibility:** ตรวจสอบให้แน่ชัดว่า DaisyUI เวอร์ชันที่ติดตั้งรองรับ Tailwind เวอร์ชันที่ใช้อยู่ — การ mismatch (เช่นใช้ daisyUI v5 กับ Tailwind v3) จะทำให้ไฟล์ CSS ไม่ถูกสร้างตามคาด. ก่อนอัพเกรดอ่าน upgrade guide ของ daisyUI (มีบันทึก breaking changes). (daisyui.com)

---

8) ข้อควรระวัง & Troubleshooting (พร้อมวิธีแก้)

- **ธีมไม่เปลี่ยน/สีหาย:** ตรวจสอบ (1) ว่า data-theme ถูกเซ็ตบน <html> หรือ scope ที่ถูกต้อง, (2) tailwind/purge ไม่ดึงไฟล์ที่มีธีมออก, (3) ถ้าคุณ override ตัวแปรโดยตรงให้แน่ใจว่า override มา **หลัง** การ import daisyUI/CSS. (daisyui.com)
- **modal ควบคุมไม่ได้ใน React:** หากใช้ checkbox-pattern ต้องแปะ checked + onChange หรือใช้ <dialog> + ref + showModal() (แนะนำ <dialog> สำหรับ accessibility). ดูตัวอย่างใน docs/StackOverflow. (daisyui.com)
- **ชื่อคลาสชนกันกับไลบรารีเก่า:** ใช้ prefix ของ daisyUI เพื่อหลีกเลี่ยง conflict (เช่น daisy-). (ถ้าต้องการ adopt แบบ incremental) (GitHub)
- **ไม่เห็น custom colors ที่เพิ่ม:** ถ้าคุณเพิ่ม color name ใหม่ลงใน theme (เช่น primary-muted) ต้องแน่ใจว่าใช้งานร่วมกับวิธีที่ daisyUI คาดหวัง — ดูตัวอย่างการเพิ่มสีใน blog ของ daisyUI. (daisyui.com)

---

9) แหล่งอ้างอิง (สำคัญ — ควรอ่าน)

- DaisyUI docs (homepage, components, utilities, config). (daisyui.com)
- daisyUI v5 release notes & config (v5 เพิ่ม include/exclude, CSS-first config). (daisyui.com)

- Colors & variables list (mapping names ⟶ --color-*). (daisyui.com)
- React integration / react-daisyui note (เพิ่ม node_modules ใน content เพื่อป้องกัน purge). (GitHub)
- Theme controller examples (docs). (daisyui.com)
- Blog / tutorial (ตัวอย่างการใช้ OKLCH และธีมใน v5). (LogRocket Blog)

---

10) แนะนำการฝึก (practical next steps)

1. สร้าง project เล็ก ๆ (Vite + React) — ติดตั้ง Tailwind + daisyUI ตามวิธี CSS-first (v5) แล้วลองสลับธีมด้วย data-theme และ ThemeSwitcher ที่เก็บใน localStorage.
2. ทำตัวอย่าง Modal ทั้ง 2 แบบ (dialog + checkbox) เพื่อเข้าใจ pattern ต่างกันและผลต่อ accessibility.
3. ทดลองสร้าง mytheme แบบ custom ด้วย OKLCH แล้วตรวจสอบ contrast/a11y (เช่นด้วย DevTools color contrast).
4. ถ้าอยากลด CSS ขนาดจริง ให้ลอง include เฉพาะ component ที่ใช้ แล้ววัดขนาดไฟล์ก่อน/หลัง.

ประวัติความเป็นมา, วิวัฒนาการ, สถิติการใช้งานปัจจุบัน และแนวโน้มในอนาคตของ daisyUI

---

1) ประวัติความเป็นมา (Who & When — ย่อ)

- **ผู้สร้าง / ผู้ริเริ่ม:** daisyUI ถูกพัฒนาโดย Pouya Saadeghi (โปรเจกต์โอเพนซอร์สบน GitHub) เพื่อเติมเต็มช่องว่างของ Tailwind — คือให้ชุด component สำเร็จรูปที่ยังคง philosophy แบบ utility-first ของ Tailwind แต่ลดงาน boilerplate ในการสร้าง UI ซ้ำ ๆ. (daisyui.com)
- **การเติบโตช่วงแรก–ต่อมา:** เริ่มเป็น plugin ยอดนิยมของชุมชน Tailwind และเติบโตอย่างรวดเร็วทั้งจากผู้ใช้ระดับบุคคลไปจนถึงโปรเจกต์โอเพนซอร์ส/เชิงพาณิชย์ — repo ได้รับ “stars” จำนวนมากและมีการออกรุ่น/อัปเดตต่อเนื่อง (changelog/บล็อกของโครงการมีบันทึกการพัฒนาอย่างละเอียด). (daisyui.com)

---

2) วิวัฒนาการเชิงเทคนิค (Major milestones & v5)

- **จาก plugin ธรรมดา ⟶ ธีมแบบ semantic + component library:** ตอนต้น daisyUI มอบชุด class/component พื้นฐาน แต่ต่อมาพัฒนาเป็นระบบธีมที่ใช้ CSS variables (semantic tokens เช่น primary, base-100, primary-content) เพื่อให้การสลับธีมทั้งโปรเจกต์เป็นไปอย่างง่ายและมีความสอดคล้อง.

- **v5 (ล่าสุด):** เป็นการอัปเดตสำคัญที่เพิ่มความสามารถเช่น *include/exclude* (โหลดเฉพาะ component ที่ต้องการเพื่อลดขนาด CSS), micro-css builds, ปรับปรุงระบบธีม และปรับปรุง accessibility/UX ของ component หลายตัว — v5 ถูกโปรโมตว่า "ลดขนาดและยืดหยุ่นขึ้น" เพื่อการใช้งานในโปรดักชัน. (daisyui.com)
- **อื่น ๆ:** มีการปรับปรุง accessibility, ปรับ default styles ตามระบบธีมใหม่, และเพิ่ม options เช่น prefix, root, include/exclude เพื่อรองรับ use-case ที่หลากหลาย. (daisyui.com)

---

3) สถิติการใช้งาน (ตัวเลขเชิงหลักฐาน — ปัจจุบัน / ล่าสุด)

(หมายเหตุ: ตัวเลขเหล่านี้มาจากแหล่งที่โปรเจกต์และหน้า package/รายงานสาธารณะเผยแพร่ — อ้างอิงด้านล่าง)

- **จำนวนโปรเจกต์ที่ใช้ (dependents / dependency graph):** รายงานจากหน้า release/v5 บอกว่า daisyUI ถูกใช้งานใน **~360,000** โครงการโอเพนซอร์ส (ตาม GitHub dependency graph / project count ที่โครงการประกาศ). (daisyui.com)
- **จำนวนการติดตั้งจาก NPM (สะสม / รายสัปดาห์):** ใน release note v5 โครงการอ้างว่า **~19 ล้าน npm installs** รวม (และประมาณ **350k weekly installs** — ตัวเลขตามประกาศของทีม). นอกจากนี้หน้า package/NPM แสดงเวอร์ชันล่าสุดและ dependents (เป็นสัญญาณว่ามีการใช้กันแพร่หลาย). (daisyui.com)
- **ดาวบน GitHub (popularity):** repo หลักมี **หลายหมื่นดาว** — หน้ารายการ repo แสดง **~38.6k** stars (ตัวเลขขึ้น/ลงตามเวลา) ซึ่งสะท้อนการยอมรับจากชุมชนอย่างสูง. (GitHub)
- **ดาวน์โหลดต่อเดือน / รายสัปดาห์ (แหล่งวิเคราะห์):** แหล่งวิเคราะห์สาธารณะเช่น BestOfJS/ตัวชี้วัด npm แสดงตัวเลขดาวน์โหลดเดือนละเป็นแสนถึงล้าน (ตัวอย่างกราฟ monthly downloads ประมาณ **1M+ ต่อเดือน** ในช่วงหลัง ๆ). (bestofjs.org)

สรุปเชิงสถิติที่น่าสนใจ: daisyUI เป็นหนึ่งใน Tailwind plugin/component library ที่ **เติบโตเร็วและถูกใช้งานอย่างกว้างขวาง** ทั้งจากดาว GitHub, รายงาน installs บน npm และการระบุว่าใช้ในหลายแสนโปรเจกต์. (daisyui.com)

---

4) แนวโน้มการใช้งานปัจจุบัน (Why it's growing now)

- **เหตุผลเชิง ecosystem:** Tailwind เองเติบโตมากในหมู่นักพัฒนาเว็บ — เมื่อ Tailwind ได้รับความนิยม ไลบรารีที่ทำให้การสร้าง UI เร็วขึ้น (เช่น daisyUI) ก็ได้รับประโยชน์จากเครือข่าย (network effect). นอกจากนี้ daisyUI เป็น **open-source, MIT license, framework-agnostic** ทำให้ adopt ง่ายทั้งในโปรเจกต์เล็ก–ใหญ่. (daisyui.com)

- **คุณสมบัติที่ตอบโจทย์โปรดักชัน:** ระบบธีม, ความสามารถในการ include/exclude, micro-css builds, และการปรับปรุง accessibility ทำให้ทีมผลิตซอฟต์แวร์พิจารณาใช้งานจริง (ไม่ใช่แค่ prototyping). v5 ออกแบบมาเพื่อตอบโจทย์การใช้งานระดับผลิต. (daisyui.com)

---

5) แนวโน้มในอนาคต (การคาดการณ์ — ชี้ชัดว่าเป็นการสรุป/คาดการณ์)

(ส่วนนี้เป็น **การวิเคราะห์เชิงเหตุผล/คาดการณ์** โดยอ้างอิงจากทิศทางของโครงการและเทรนด์ของ ecosystem — ไม่ใช่ตัวเลขที่โครงการประกาศ)

- **การเติบโตต่อเนื่องในระยะสั้น–กลาง:** คาดว่า daisyUI จะยังโตต่อ เพราะ (1) Tailwind ยังคงได้รับความนิยม, (2) daisyUI มี community momentum (stars, downloads), (3) การพัฒนา v5 ช่วยลดขนาดและรองรับ use-case ระดับโปรดักชัน ⟶ ทำให้ adoption ในบริษัท/ทีมเพิ่มขึ้น. (นี่เป็นการสรุปเหตุผล ไม่ใช่ตัวเลขทางการ).
- **ทิศทางฟีเจอร์:** คาดว่าโครงการจะเน้นเรื่อง *performance (micro builds)*, *theming sophistication* (OKLCH, better color tooling), *accessibility*, และ *integration with design-system/workflow* (เช่น Storybook, tokens) — เหตุผล: เหล่านี้เป็นหัวข้อ roadmap และสิ่งที่ชุมชนอยากเห็น. (daisyui.com)
- **การแข่งขัน & ที่ว่างทางตลาด:** daisyUI จะยังแข่งขันกับตัวเลือกอื่น (เช่น Tailwind UI — แบบพรีเมียม, shadcn/ui, Preline) แต่จุดแข็งคือ *ฟรี, framework-agnostic, มีธีมเยอะ* — ทำให้มีตลาดเฉพาะตัวอยู่. ถ้าต้องการใช้ UI แบบออกแบบละเอียดตามแบรนด์มาก ๆ ทีมอาจใช้ daisyUI เป็น base แล้ว extend ต่อ. (daisyui.com)

---

6) ข้อจำกัดที่อาจกระทบแนวโน้ม

- **ความจำเป็นต้อง custom ลึก ๆ:** หากโปรเจกต์ต้องการ branding ที่ละเอียดทุกพิกเซล บางทีมอาจหลีกเลี่ยงการใช้ component ที่ opinionated มาก และเลือกสร้าง component เองหรือใช้ paid Tailwind UI/Design system.
- **ความเข้ากันกับ tooling เก่า / purge:** โปรเจกต์ที่ใช้การ generate class แบบ runtime ต้องตั้งค่า content/safelist ให้ถูกต้อง มิฉะนั้น purge อาจตัด class ที่ daisyUI สร้างออกได้ — เป็นข้อปฏิบัติที่ต้องระวังเมื่อนำไปใช้จริง. (daisyui.com)

---

7) ข้อสรุปสั้น ๆ (Executive summary)

- daisyUI เกิดจากความต้องการลด boilerplate ของ Tailwind และเติบโตมาเป็น **หนึ่งใน Tailwind component libraries ที่ได้รับความนิยมสูงสุด** (ดาว GitHub เป็นหมื่น ๆ, npm installs เป็นหลักหลายล้าน, ถูกใช้งานในแสนกว่าโปรเจกต์). (GitHub)

- วิวัฒนาการล่าสุด (v5) โฟกัสที่ *performance (micro builds, include/exclude)*, *theming improvements* และ *accessibility* — ทำให้เครื่องมือนี้พร้อมสำหรับการใช้งานระดับ production มากขึ้น. (daisyui.com)
- แนวโน้มในอนาคต: คาดว่า **เติบโตต่อ** ร่วมกับ ecosystem ของ Tailwind — แต่การใช้งานจริงจะขึ้นกับความต้องการ customization ของแต่ละทีมและการจัดการเรื่อง performance/purge. (การคาดการณ์นี้เป็นการวิเคราะห์ ไม่ใช่ตัวเลขทางการ). (daisyui.com)

---

8) แหล่งอ้างอิงสำคัญ (ตรวจสอบต่อ)

1. daisyUI official site — v5 release notes (ตัวเลข installs / projects / v5 features). (daisyui.com)
2. GitHub repo (saadeghi/daisyui) — stars, repo activity. (GitHub)
3. NPM package page — เวอร์ชันล่าสุด, dependents, publish history. (npm)
4. BestOfJS / package analytics — downloads charts, monthly download figures. (bestofjs.org)
5. Blog / changelog / roadmap ของ daisyUI — notes เกี่ยวกับ v5, roadmap, changelog รายละเอียดฟีเจอร์. (daisyui.com)

DaisyUI คืออะไร?

DaisyUI คือ **Tailwind CSS plugin** ที่เพิ่ม *component classes* (เช่น btn, card, navbar, modal) และ *theme system* สำเร็จรูปเข้าไปใน Tailwind CSS

- ปกติ **Tailwind CSS** เป็น **utility-first framework** ⟶ คุณต้องใช้คลาสเล็ก ๆ (px-4, bg-blue-500, rounded-lg) มาประกอบเองเพื่อสร้างปุ่ม, card, modal
- DaisyUI ทำหน้าที่ **ห่อ utility classes** เหล่านั้นให้เป็น **semantic component class** ⟶ เช่น btn btn-primary = ปุ่มสวยพร้อมสีหลัก โดยไม่ต้องเขียนยูทิลิตี้ยาว ๆ เอง

 นั่นหมายความว่า DaisyUI ไม่ได้แทนที่ Tailwind แต่ทำงาน **เป็นส่วนเสริม** เพื่อเร่งการสร้าง UI ให้เร็วขึ้น โดยยังใช้พื้นฐานของ Tailwind อยู่

DaisyUI แก้ปัญหาอะไรของ Tailwind?

1. ลดงานซ้ำซ้อน (Boilerplate)

- **Tailwind เพียว ๆ:** การสร้างปุ่มมาตรฐานหนึ่งปุ่มอาจต้องใช้โค้ดแบบนี้:
- <button class="px-4 py-2 bg-blue-500 hover:bg-blue-600 text-white font-semibold rounded-lg shadow">
- Click me

- </button>
- **DaisyUI:** ใช้เพียง
- <button class="btn btn-primary">Click me</button>
- ผลลัพธ์เหมือนกัน แต่ DaisyUI ใช้โค้ดสั้นกว่า อ่านง่ายกว่า

---

2. ระบบธีม (Themes)

- Tailwind เองไม่มีระบบธีม built-in ⟶ ถ้าจะทำ dark/light mode ต้องจัดการสีเอง
- DaisyUI มี **กว่า 30+ themes สำเร็จรูป** และระบบ custom theme (ผ่าน CSS variables เช่น --p, --pc, --b1, --bc) ⟶ ทำให้เปลี่ยนธีมได้ทั้งเว็บแบบทันที
- ตัวอย่าง:
- <html data-theme="dark">...</html>
- <html data-theme="cupcake">...</html>
  ⟶ UI ทั้งเว็บเปลี่ยนโทนสีได้ทันที

---

3. ความสม่ำเสมอ (Consistency)

- ในโปรเจกต์ใหญ่ ถ้า dev แต่ละคนเขียน utility classes เอง อาจไม่สม่ำเสมอ
- DaisyUI ใช้ **class มาตรฐานกลาง** (btn, card, alert, badge) ทำให้ทุกคนใช้เหมือนกัน ⟶ UI มีมาตรฐานเดียวกัน

---

4. เวลาในการพัฒนา (Speed)

- นักพัฒนาไม่ต้องออกแบบ component UI ตั้งแต่ศูนย์
- DaisyUI มี component พร้อมใช้ เช่น **ปุ่ม, navbar, modal, drawer, table, form** ⟶ ทำ prototype ได้เร็วมาก และนำไปใช้จริงในโปรดักชันได้ทันที

---

5. คงความยืดหยุ่นของ Tailwind

- ถึงจะใช้ DaisyUI ก็ยังสามารถเพิ่ม/override ด้วย utility classes ของ Tailwind ได้
- เช่น btn btn-primary px-8 ⟶ DaisyUI สร้างปุ่มหลัก + Tailwind ปรับ padding เพิ่ม

สรุปสั้น ๆ:

DaisyUI = **“Tailwind + Ready-made UI Kit + Theme System”**

มันแก้ปัญหาของ Tailwind ที่ **เขียน class ยาวเกินไป, ไม่มี component สำเร็จรูป, ไม่มีระบบธีมในตัว** และช่วยให้ dev ทำงานเร็วขึ้นแต่ยังยืดหยุ่นสูง

---

6 โปรแกรม โดยแบ่งเป็น

- **พื้นฐาน (Basic) 3 โปรแกรม** ⟶ เน้นการใช้ DaisyUI + Tailwind เบื้องต้น
- **ประยุกต์ (Applied) 3 โปรแกรม** ⟶ สร้างแอปเล็ก ๆ ที่ใช้ component ของ DaisyUI

ทั้งหมดจะมี **โครงสร้างโปรเจกต์, ไฟล์โค้ด, คำอธิบาย, ผลการรัน (UI ที่ได้)**

☐ **ส่วนที่ 1: โปรแกรมพื้นฐาน (Basic Examples)**

☐ **ตัวอย่างที่ 1: ปุ่มพื้นฐาน (Button)**

โครงสร้างไฟล์

```
daisyui-basic-button/
├── index.html
├── tailwind.config.js
└── package.json
```

index.html

```
<!DOCTYPE html>
<html lang="en" data-theme="light">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>DaisyUI Button</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <script>
    tailwind.config = {
      plugins: [daisyui],
    }
  </script>
</head>
<body class="flex items-center justify-center h-screen">
  <button class="btn btn-primary">Click Me</button>
</body>
</html>
```

คำอธิบาย

- ใช้ DaisyUI btn btn-primary ⟶ ปุ่มสีน้ำเงินพร้อม hover effect
- ไม่ต้องเขียน utility class ยาว ๆ

ผลการรัน

ปุ่มสีน้ำเงินกลมมนที่เขียนว่า **Click Me** อยู่ตรงกลางจอ

☐ ตัวอย่างที่ 2: การ์ด (Card)

index.html

```
<!DOCTYPE html>
<html lang="en" data-theme="cupcake">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>DaisyUI Card</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <script>
    tailwind.config = { plugins: [daisyui] }
  </script>
</head>
<body class="flex items-center justify-center h-screen">
  <div class="card w-96 bg-base-100 shadow-xl">
    <figure><img src="https://placekitten.com/400/200" alt="Cat"/></figure>
    <div class="card-body">
      <h2 class="card-title">Cute Cat!</h2>
      <p>This is a lovely cat photo.</p>
      <div class="card-actions justify-end">
        <button class="btn btn-secondary">Like</button>
      </div>
    </div>
  </div>
</body>
</html>
```

คำอธิบาย

- card ของ DaisyUI มี header + body + footer ให้
- btn-secondary = ปุ่มโทนรอง (สีชมพู/ม่วงตาม theme cupcake)

ผลการรัน

แสดงการ์ดขนาดกลาง มีรูปแมวด้านบน, ข้อความ, และปุ่ม **Like**

☐ **ตัวอย่างที่ 3: Navbar**

index.html

```
<!DOCTYPE html>
<html lang="en" data-theme="dark">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>DaisyUI Navbar</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <script>
    tailwind.config = { plugins: [daisyui] }
  </script>
</head>
<body>
  <div class="navbar bg-base-200">
    <div class="flex-1">
      <a class="btn btn-ghost normal-case text-xl">MyApp</a>
    </div>
    <div class="flex-none gap-2">
      <button class="btn">Login</button>
    </div>
  </div>
</body>
</html>
```

คำอธิบาย

- ใช้ navbar component ⟶ ได้ layout พร้อม background
- ปุ่ม Login อยู่ด้านขวา

ผลการรัน

แถบ navbar มืด (dark theme), มีโลโก้ **MyApp** ทางซ้าย และปุ่ม **Login** ทางขวา

☐ **ส่วนที่ 2: โปรแกรมประยุกต์ (Applied Examples)**

☐ **ตัวอย่างที่ 4: ระบบสลับธีม (Theme Switcher)**