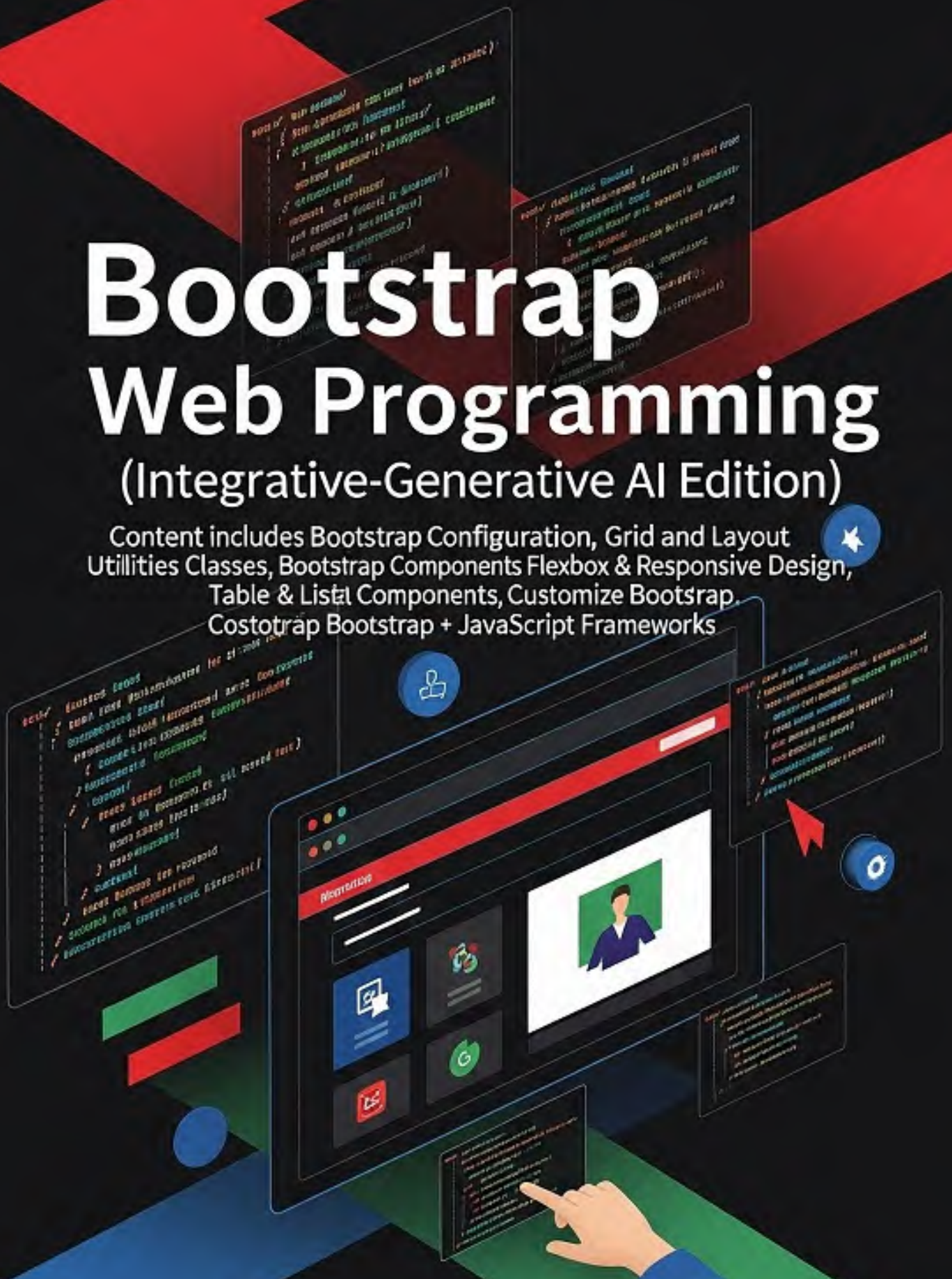
Bootstrap
Web Programming
(Integrative-Generative AI Edition)

Content includes Bootstrap Configuration, Grid and Layout
Utillities Classes, Bootstrap Components Flexbox & Responsive Design,
Table & List Components, Customize Bootsrap,
Costotrap Bootstrap + JavaScript Frameworks



Student Price Book Center

คำนำ

หนังสือเล่มนี้จัดทำขึ้นเพื่อเป็นคู่มือในการเรียนรู้และใช้งาน **Bootstrap** อย่างเป็นระบบ โดยออกแบบเนื้อหาให้ครอบคลุมตั้งแต่ระดับพื้นฐานไปจนถึงระดับสูง เหมาะสำหรับผู้เริ่มต้นที่ไม่มีพื้นฐานมาก่อน ตลอดจนผู้มีประสบการณ์ที่ต้องการต่อยอดและพัฒนาทักษะให้ลึกซึ้งยิ่งขึ้น

เนื้อหาแบ่งออกเป็น 3 ระดับ ได้แก่

- **ขั้นพื้นฐาน (Beginner)** ครอบคลุมความรู้เบื้องต้นเกี่ยวกับ Bootstrap การติดตั้งระบบ การจัดการเลย์เอาต์ด้วย Grid System และคลาสช่วยเหลือ (Utility Classes) ที่จำเป็นในการพัฒนาเว็บเพจที่ตอบสนองได้ทุกขนาดหน้าจอ
- **ขั้นกลาง (Intermediate)** มุ่งเน้นการใช้งานคอมโพเนนต์ต่าง ๆ เช่น Navbar, Button, Card, Form ตลอดจนการจัดวางหน้าเว็บด้วย Flexbox และการใช้งานตารางข้อมูลและกลุ่มรายการ
- **ขั้นสูง (Advanced)** เจาะลึกการใช้งาน Bootstrap ร่วมกับ JavaScript เช่น Carousel, Tooltip, Modal รวมถึงการปรับแต่งด้วย SCSS และการบูรณาการกับ JavaScript Frameworks เช่น jQuery, React และ Vue

จุดเด่นของหนังสือเล่มนี้คือการผสาน **Generative AI** เข้ากับการจัดเรียงเนื้อหาอย่างเป็นลำดับ โดยเน้นการสื่อสารที่กระชับ ชัดเจน เข้าใจง่าย ลดความเยิ่นเย้อของคำอธิบาย และเน้นให้ผู้อ่านสามารถ “ลงมือทำได้จริง” ผ่านตัวอย่างที่จับต้องได้ เหมาะสำหรับการเรียนรู้ด้วยตนเอง การสอนในชั้นเรียน หรือการประยุกต์ใช้ในการพัฒนาเว็บไซต์จริง

หวังเป็นอย่างยิ่งว่าหนังสือเล่มนี้จะเป็นประโยชน์ต่อผู้อ่านในการเรียนรู้ Bootstrap อย่างมั่นใจ และสามารถประยุกต์ใช้ในงานพัฒนาเว็บในโลกปัจจุบันที่เปลี่ยนแปลงอย่างรวดเร็ว

ด้วยความปรารถนาดี

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 1
ความรู้เบื้องต้น
(Basic of Bootstrap)

เนื้อหา

- Bootstrap คืออะไร?
- Bootstrap เหมาะกับใคร?
- ข้อมูลเพิ่มเติมเกี่ยวกับ Bootstrap
- การติดตั้ง Bootstrap
- โครงสร้างพื้นฐานของ Bootstrap

Bootstrap คืออะไร?

Bootstrap เป็นเฟรมเวิร์ก CSS ยอดนิยมที่ช่วยให้การพัฒนาเว็บไซต์ง่ายขึ้น ด้วยเครื่องมือสำเร็จรูปสำหรับออกแบบ UI ที่สวยงามและรองรับทุกอุปกรณ์ (Responsive Design) **Bootstrap** ช่วยให้นักพัฒนาเว็บสามารถสร้างเว็บที่มีดีไซน์สวยงามและตอบสนองทุกอุปกรณ์ (Responsive) ได้ง่ายขึ้น โดยไม่ต้องเขียน CSS เองทั้งหมด

คุณสมบัติเด่นของ Bootstrap

☐ **Responsive Design** – รองรับทุกอุปกรณ์ (มือถือ, แท็บเล็ต, คอมพิวเตอร์)

☐ **Grid System** – ระบบ 12 คอลัมน์ที่ช่วยจัดวางเลย์เอาต์ได้สะดวก

☐ **Component สำเร็จรูป** – ปุ่ม (Button), แบบฟอร์ม (Form), เมนู (Navbar), โมดอล (Modal), การ์ด (Card) ฯลฯ

☐ **Utility Classes** – ใช้คลาสสำเร็จรูปช่วยจัดการ Margin, Padding, สี, ขนาดตัวอักษร ฯลฯ

☐ **JavaScript Plugin** – เช่น Modal, Carousel, Dropdown ที่ไม่ต้องเขียนโค้ด JS เอง

เวอร์ชันของ Bootstrap

- **Bootstrap 3** – รองรับ jQuery, ใช้ float เป็นหลัก
- **Bootstrap 4** – ใช้ Flexbox, ปรับปรุง Grid System
- **Bootstrap 5 (ปัจจุบัน)** – ตัด jQuery ออก, ใช้ CSS Grid, ปรับแต่งได้ง่ายขึ้น

ตัวอย่างการใช้งาน Bootstrap

1. วิธีใช้ Bootstrap ผ่าน CDN

เพิ่มโค้ดนี้ใน <head> ของ HTML

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>ตัวอย่าง Bootstrap</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <div class="container text-center">
        <h1 class="text-primary">Hello, Bootstrap!</h1>
        <button class="btn btn-success">คลิกเลย</button>
    </div>
</body>
</html>
```

Bootstrap เหมาะกับใคร?

- นักพัฒนาเว็บที่ต้องการสร้างเว็บให้สวยงามและใช้งานง่าย
- ผู้ที่ต้องการทำเว็บให้ **Responsive** โดยไม่ต้องเขียน CSS เอง
- นักพัฒนาที่ต้องการลดเวลาทำงานและใช้ Component สำเร็จรูป

คุณสนใจใช้ Bootstrap กับโปรเจ็กต์อะไรครับ? □

ประวัติความเป็นมาของ Bootstrap

Bootstrap เป็น **CSS Framework** ที่ถูกพัฒนาโดยทีมงานของ Twitter เพื่อช่วยให้นักพัฒนาสามารถสร้างเว็บไซต์ที่สวยงามและรองรับทุกอุปกรณ์ได้ง่ายขึ้น

จุดเริ่มต้นของ Bootstrap

□ **ปี 2010** – Mark Otto และ Jacob Thornton สองวิศวกรของ Twitter ได้พัฒนา **"Twitter Blueprint"** ซึ่งเป็นระบบออกแบบภายในของ Twitter

□ **ปี 2011** – Twitter ตัดสินใจ **เปิดซอร์ส** เฟรมเวิร์กนี้และเปลี่ยนชื่อเป็น **Bootstrap** เปิดตัวครั้งแรกในวันที่ **19 สิงหาคม 2011**

□ **ปี 2012** – Bootstrap กลายเป็นโปรเจ็กต์ยอดนิยมบน GitHub

พัฒนาการของ Bootstrap ในแต่ละเวอร์ชัน

เวอร์ชัน	ปีที่เปิดตัว	คุณสมบัติเด่น
Bootstrap 2	2012	รองรับ Responsive Design ครั้งแรก
Bootstrap 3	2013	ปรับปรุง Grid System และใช้ Flexbox บางส่วน
Bootstrap 4	2018	เปลี่ยนมาใช้ Flexbox เต็มรูปแบบ, รองรับ SASS
Bootstrap 5	2021	ตัด jQuery ออก, ใช้ CSS Grid, ปรับปรุง UI

ปัจจุบัน

Bootstrap เป็น **CSS Framework** ที่ได้รับความนิยมมากที่สุดในโลก มีการใช้งานอย่างแพร่หลายในเว็บไซต์และแอปพลิเคชันมากมาย

คุณสนใจรายละเอียดของเวอร์ชันไหนเป็นพิเศษไหมครับ? □

ข้อมูลเพิ่มเติมเกี่ยวกับ Bootstrap

1. จุดเด่นของ Bootstrap

- □ **ใช้งานง่าย** – มีคลาส CSS สำเร็จรูปช่วยลดเวลาเขียนโค้ด
- □ **รองรับ Responsive Design** – ปรับเลย์เอาต์อัตโนมัติตามขนาดหน้าจอ
- □ **มาพร้อม UI Components มากมาย** – เช่น ปุ่ม (Button), การ์ด (Card), เมนู (Navbar), ฟอร์ม (Form), โมดอล (Modal)
- □ **รองรับการปรับแต่ง** – ใช้ CSS หรือ SASS เพื่อปรับแต่งเพิ่มเติม
- ⚡ **รองรับ JavaScript Plugin** – เช่น Tooltip, Carousel, Dropdown โดยใช้ Vanilla JavaScript

---

2. รายละเอียดของเวอร์ชันต่าง ๆ

□ **Bootstrap 2 (2012)**

- เพิ่มระบบ **Responsive Design** ครั้งแรก
- ยังใช้ Grid System แบบ **Fixed Layout**

□ **Bootstrap 3 (2013)**

- ปรับปรุง Grid System ให้เป็น **Mobile-first** (ออกแบบให้เหมาะกับมือถือก่อน)
- ใช้ **Flexbox บางส่วน**
- เปลี่ยน UI ให้ดูสะอาดขึ้น

□ **Bootstrap 4 (2018)**

- ใช้ **Flexbox เต็มรูปแบบ**
- เพิ่ม **SASS** (แทน LESS) ทำให้ปรับแต่งได้ง่ายขึ้น
- รองรับ **Cards Component** แทน Panel, Thumbnail และ Well

☐ **Bootstrap 5 (2021 - ปัจจุบัน)**

- ตัด jQuery ออก ใช้ **Vanilla JavaScript**
- รองรับ **CSS Grid** และปรับปรุง Flexbox
- เพิ่มฟีเจอร์ **Utilities API** ให้กำหนดค่า CSS ได้ง่ายขึ้น
- รองรับ **RTL (Right-to-Left)** เช่น ภาษาอาหรับ

---

3. Bootstrap Grid System (12 คอลัมน์)

Bootstrap ใช้ระบบ **Grid แบบ 12 คอลัมน์** ซึ่งช่วยให้สามารถจัดวางเลย์เอาต์ได้อย่างยืดหยุ่น

ตัวอย่าง Grid System

```
<div class="container">
    <div class="row">
        <div class="col-md-6 bg-primary text-white">50%</div>
        <div class="col-md-6 bg-secondary text-white">50%</div>
    </div>
</div>
```

☐ *ในตัวอย่างนี้ col-md-6 หมายถึง ใช้พื้นที่ 6 คอลัมน์จาก 12 คอลัมน์ในหน้าจอขนาดกลางขึ้นไป*

---

4. ตัวอย่างการใช้งาน Bootstrap

☐ **เพิ่ม Bootstrap ผ่าน CDN**

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
```

☐ **ปุ่มสวย ๆ ด้วย Bootstrap**

```
<button class="btn btn-primary">คลิกเลย</button>
<button class="btn btn-success">Success</button>
<button class="btn btn-danger">Danger</button>
```

☐ **Navbar (เมนู)**

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">My Website</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav">
```

```
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarNav">
      <ul class="navbar-nav">
        <li class="nav-item"><a class="nav-link active" href="#">Home</a></li>
        <li class="nav-item"><a class="nav-link" href="#">About</a></li>
      </ul>
    </div>
  </div>
</nav>
```

---

5. การใช้งาน Bootstrap กับ JavaScript

Bootstrap มี **JavaScript Plugins** ที่ช่วยเพิ่มฟังก์ชัน เช่น โมดอล (Modal), ทูลทิป (Tooltip), คารูเซล (Carousel)

ตัวอย่าง Modal (ป๊อปอัป)

```
<button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-target="#myModal">
  เปิดโมดอล
</button>

<div class="modal fade" id="myModal">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modal Title</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal"></button>
      </div>
      <div class="modal-body">
        นี่คือเนื้อหาของโมดอล
      </div>
    </div>
  </div>
</div>
```

---

6. การติดตั้ง Bootstrap แบบ Offline

หากต้องการใช้งานแบบออฟไลน์ สามารถดาวน์โหลด Bootstrap ได้จาก

- https://getbootstrap.com

หรือใช้คำสั่งใน **Node.js**

npm install bootstrap

---

7. Bootstrap ใช้กับอะไรได้บ้าง?

- ใช้กับ **HTML, CSS, JavaScript**
- ใช้กับ **JSP, PHP, React, Angular, Vue.js** ได้
- ใช้ร่วมกับ **PrimeFaces (JSF), jQuery, AJAX** ได้

---

8. Bootstrap กับ UI Framework อื่น ๆ

Framework	จุดเด่น
Bootstrap	ใช้งานง่าย, มี Component มากมาย
Tailwind CSS	ปรับแต่งได้สูง, ใช้ Utility Classes
Materialize CSS	ใช้ดีไซน์แนว Material Design
Foundation	เหมาะกับ Responsive Web Design

สรุป

- **Bootstrap คือเฟรมเวิร์ก CSS ยอดนิยม** ที่ช่วยให้ออกแบบเว็บไซต์ได้ง่ายและสวยงาม
- มี **Grid System, Component สำเร็จรูป, และ JavaScript Plugins**
- ใช้ได้ทั้งแบบ **CDN และ Offline**
- ปัจจุบัน Bootstrap 5 เป็นเวอร์ชันล่าสุด

การติดตั้ง Bootstrap

Bootstrap สามารถติดตั้งและใช้งานได้หลายวิธี เช่น

1. **ใช้ผ่าน CDN (ง่ายที่สุด)**
2. **ดาวน์โหลดไฟล์และใช้งานแบบออฟไลน์**
3. **ติดตั้งผ่าน npm (สำหรับนักพัฒนาเว็บที่ใช้ Node.js)**

---

1. การติดตั้ง Bootstrap ผ่าน CDN (ง่ายที่สุด)

CDN (Content Delivery Network) ช่วยให้ใช้งาน Bootstrap ได้โดยไม่ต้องดาวน์โหลดไฟล์

ขั้นตอน

1. เปิด **ไฟล์ HTML** ของคุณ
2. เพิ่มลิงก์ CDN ของ Bootstrap ลงใน <head>
3. เพิ่มโค้ด HTML และใช้คลาสของ Bootstrap

ตัวอย่าง

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap ผ่าน CDN</title>
    <!-- ลิงก์ CSS ของ Bootstrap -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>

<div class="container text-center">
    <h1 class="text-primary">สวัสดี Bootstrap!</h1>
    <button class="btn btn-success">คลิกเลย</button>
</div>

<!-- ลิงก์ JavaScript ของ Bootstrap (จำเป็นสำหรับบางฟังก์ชัน เช่น Modal, Dropdown) -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

- **ข้อดี**: ง่าย รวดเร็ว ไม่ต้องดาวน์โหลด
- **ข้อเสีย**: ต้องใช้ **อินเทอร์เน็ต** ทุกครั้งที่โหลดเว็บ

---

2. การติดตั้ง Bootstrap แบบออฟไลน์ (ดาวน์โหลดไฟล์)

ถ้าไม่ต้องการใช้ CDN สามารถดาวน์โหลด Bootstrap และใช้งานแบบออฟไลน์ได้

ขั้นตอน

1. ไปที่ https://getbootstrap.com
2. คลิก **"Download"** แล้วเลือก **Compiled CSS and JS**
3. แตกไฟล์ ZIP และคัดลอกโฟลเดอร์ bootstrap-5.x.x-dist ไปยังโปรเจ็กต์ของคุณ
4. ลิงก์ไฟล์ **CSS และ JS** ไปยัง HTML

ตัวอย่าง

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap แบบ Offline</title>
    <!-- ลิงก์ไปยังไฟล์ Bootstrap ที่ดาวน์โหลด -->
    <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">
</head>
<body>

<div class="container">
    <h1 class="text-success">Hello, Bootstrap Offline!</h1>
</div>

<!-- ใส่ JavaScript เพื่อให้ฟังก์ชัน Bootstrap ทำงาน -->
<script src="bootstrap/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

☐ **ข้อดี**: ใช้ได้แม้ไม่มีอินเทอร์เน็ต

☐ **ข้อเสีย**: ต้องดาวน์โหลดไฟล์และจัดการเอง

---

3. การติดตั้ง Bootstrap ด้วย npm (สำหรับนักพัฒนา Node.js)

ถ้าคุณใช้ **Node.js** หรือ **Webpack** สามารถติดตั้ง Bootstrap ผ่าน **npm** ได้

ขั้นตอน

1. ติดตั้ง Bootstrap ด้วยคำสั่ง
2. npm install bootstrap
3. นำเข้าไฟล์ Bootstrap ในโค้ด **CSS หรือ JavaScript**

ตัวอย่างการใช้ใน CSS

@import 'bootstrap/dist/css/bootstrap.min.css';

ตัวอย่างการใช้ใน JavaScript

import 'bootstrap/dist/css/bootstrap.min.css';

import 'bootstrap/dist/js/bootstrap.bundle.min.js';

- **ข้อดี**: จัดการเวอร์ชันง่าย, ใช้ร่วมกับ React, Angular, Vue ได้
- **ข้อเสีย**: ต้องติดตั้ง **Node.js** และ npm

---

4. การตรวจสอบว่า Bootstrap ติดตั้งสำเร็จหรือไม่

หลังจากติดตั้ง Bootstrap แล้ว ให้ลองสร้าง **ปุ่ม Bootstrap**

<button class="btn btn-primary">ทดสอบปุ่ม</button>

ถ้าปุ่มมีดีไซน์สวยงาม แสดงว่า Bootstrap ทำงานถูกต้อง

สรุปวิธีการติดตั้ง Bootstrap

วิธีติดตั้ง	ข้อดี	ข้อเสีย
CDN	ง่าย ไม่ต้องติดตั้ง	ใช้ไม่ได้ถ้าไม่มีอินเทอร์เน็ต
ดาวน์โหลดไฟล์ (Offline)	ใช้ได้ทุกที่	ต้องดาวน์โหลดและอัปเดตเอง
npm	เหมาะกับโปรเจ็กต์ใหญ่ ใช้ร่วมกับ React, Vue	ต้องใช้ Node.js และ npm

---

โครงสร้างพื้นฐานของ Bootstrap

Bootstrap มีโครงสร้างหลักที่ช่วยให้นักพัฒนาสร้างเว็บไซต์ที่ดูดีและรองรับทุกอุปกรณ์ได้ง่ายขึ้น โดยโครงสร้างพื้นฐานของ Bootstrap ประกอบด้วย **5 ส่วนหลัก** ได้แก่

1. **Container (คอนเทนเนอร์)**
2. **Grid System (ระบบกริด)**
3. **Typography (รูปแบบข้อความ)**
4. **Components (องค์ประกอบสำเร็จรูป)**
5. **Utilities (คลาสช่วยเหลือ)**

---

1. Container (คอนเทนเนอร์)

คอนเทนเนอร์เป็นส่วนที่ใช้ **กำหนดขอบเขตของเนื้อหา** เพื่อให้เลย์เอาต์ของเว็บดูเป็นระเบียบ

ประเภทของ Container

ชนิด	คำอธิบาย
.container	ขนาดคงที่ตามจอแสดงผล (Responsive)
.container-fluid	กว้างเต็มจอเสมอ

ตัวอย่าง

```
<div class="container bg-light p-3">
    <h3>คอนเทนเนอร์แบบกำหนดขนาด</h3>
</div>

<div class="container-fluid bg-secondary text-white p-3">
    <h3>คอนเทนเนอร์แบบเต็มจอ</h3>
</div>
```

☐ **ผลลัพธ์**:

- .container จะมีขอบเขตตามขนาดหน้าจอ
- .container-fluid จะยืดเต็มหน้าจอ

---

2. Grid System (ระบบกริด)

Bootstrap ใช้ **Grid แบบ 12 คอลัมน์** เพื่อช่วยจัดเลย์เอาต์ของหน้าเว็บ

ตัวอย่างการใช้ Grid

```
<div class="container">
    <div class="row">
        <div class="col-md-4 bg-primary text-white p-3">คอลัมน์ 1</div>
        <div class="col-md-4 bg-success text-white p-3">คอลัมน์ 2</div>
        <div class="col-md-4 bg-danger text-white p-3">คอลัมน์ 3</div>
    </div>
</div>
```

☐ **ผลลัพธ์**:

- เมื่อหน้าจอมีขนาดกลางขึ้นไป (md) แต่ละคอลัมน์จะกว้าง 4/12 ของพื้นที่ทั้งหมด

---

3. Typography (รูปแบบข้อความ)

Bootstrap มีสไตล์ตัวอักษรที่ใช้งานง่าย

ตัวอย่าง

```
<div class="container">
```

```
    <h1>หัวข้อหลัก (H1)</h1>
    <p class="lead">ข้อความนำ (Lead text) ที่มีขนาดใหญ่กว่าปกติ</p>
    <p>นี่คือข้อความปกติ</p>
    <p class="text-muted">นี่คือข้อความสีจาง</p>
</div>
```

☐ **ผลลัพธ์**:

- lead ทำให้ข้อความใหญ่ขึ้น
- text-muted ทำให้สีจางลง

---

4. Components (องค์ประกอบสำเร็จรูป)

Bootstrap มีองค์ประกอบ UI สำเร็จรูป เช่น **ปุ่ม (Button), การ์ด (Card), Navbar, โมดอล (Modal)** เป็นต้น

ตัวอย่างปุ่ม

```
<button class="btn btn-primary">ปุ่มหลัก</button>
<button class="btn btn-success">ปุ่มสำเร็จ</button>
<button class="btn btn-danger">ปุ่มอันตราย</button>
```

☐ **ผลลัพธ์**: ปุ่มที่มีสีแตกต่างกันตามประเภท

---

5. Utilities (คลาสช่วยเหลือ)

Utilities เป็นคลาสที่ช่วยปรับแต่ง **ระยะห่าง, สี, การจัดวาง, ขนาดตัวอักษร** เป็นต้น

ตัวอย่าง

```
<div class="container">
    <p class="text-center text-warning">ข้อความตรงกลางสีเหลือง</p>
    <p class="bg-dark text-white p-3">พื้นหลังสีดำ ข้อความสีขาว</p>
    <p class="m-5 p-3 border border-danger">ขอบสีแดง + ระยะห่าง 5</p>
</div>
```

☐ **ผลลัพธ์**:

- text-center ทำให้ข้อความอยู่ตรงกลาง
- bg-dark text-white ทำให้พื้นหลังดำและข้อความขาว
- border border-danger ใส่ขอบสีแดง

สรุปโครงสร้างพื้นฐานของ Bootstrap

โครงสร้าง	คำอธิบาย	ตัวอย่าง

โครงสร้าง	คำอธิบาย	ตัวอย่าง
Container	ใช้กำหนดขอบเขตของเนื้อหา	.container, .container-fluid
Grid System	ระบบ 12 คอลัมน์สำหรับจัดวางเลย์เอาต์	.row, .col-md-6
Typography	รูปแบบตัวอักษร	.lead, .text-muted
Components	องค์ประกอบสำเร็จรูป เช่น ปุ่ม, Navbar, Card	.btn, .card
Utilities	คลาสช่วยเหลือ เช่น สี, ระยะห่าง, การจัดวาง	.text-center, .p-3, .m-5

สรุป

Bootstrap คือเฟรมเวิร์ก CSS ที่ได้รับความนิยมอย่างมาก เพราะช่วยให้การพัฒนาเว็บไซต์เป็นเรื่องง่ายและรวดเร็วขึ้น ด้วยเครื่องมือสำเร็จรูปสำหรับออกแบบ UI ที่สวยงามและรองรับทุกขนาดหน้าจอ (Responsive Design) นักพัฒนาไม่จำเป็นต้องเขียน CSS เองทั้งหมด เนื่องจาก Bootstrap มีระบบ Grid 12 คอลัมน์สำหรับจัด Layout, Container สำหรับกำหนดขอบเขตเนื้อหา และคลาส Utilities สำหรับจัดการระยะห่าง สี ขนาดตัวอักษร และการจัดวางองค์ประกอบ นอกจากนี้ยังสามารถติดตั้งได้หลากหลายวิธี เช่น ผ่าน CDN, ดาวน์โหลดไฟล์, หรือใช้ผ่าน npm/yarn เพื่อความสะดวกในการพัฒนาแบบมืออาชีพ

ในระดับที่สูงขึ้น Bootstrap ยังมี Components สำเร็จรูป เช่น Navbar, Buttons, Forms, Cards, Modals, และ Alerts ซึ่งช่วยให้ออกแบบ UI ได้อย่างมืออาชีพ พร้อมทั้งสนับสนุน Flexbox สำหรับการจัดวางองค์ประกอบอย่างยืดหยุ่น มีคลาสที่ช่วยควบคุมการแสดงผลตามขนาดหน้าจอ และรองรับการจัดการตารางและรายการแบบ List Group สำหรับขั้นสูง Bootstrap ยังมี JavaScript Components เช่น Carousel, Tooltip, Accordion และสามารถปรับแต่งด้วย SCSS, ใช้ร่วมกับโหมดมืด (Dark Mode) และทำงานร่วมกับเฟรมเวิร์ก JavaScript เช่น jQuery, React (ผ่าน React-Bootstrap) และ Vue (ผ่าน BootstrapVue) ได้อีกด้วย ทั้งนี้มีแหล่งเรียนรู้ออนไลน์มากมาย เช่น เอกสารทางการ, W3Schools และตัวอย่างโค้ดให้ศึกษาเพิ่มเติม.

บทที่ 2

กริดและเลเอาท์

(Grid and Layout)

เนื้อหา

- Grid System (ระบบ 12 คอลัมน์) ใน Bootstrap
- ข้อมูลเพิ่มเติมเกี่ยวกับ Bootstrap Grid System
- ตัวอย่างโค้ด Bootstrap Grid System แบบเต็ม
- Container ใน Bootstrap คืออะไร?
- Row & Column ใน Bootstrap
- ข้อมูลเพิ่มเติมเกี่ยวกับ Row และ Column ใน Bootstrap
- Breakpoints & Responsive Design
- ข้อมูลเพิ่มเติมเกี่ยวกับ Breakpoints & Responsive Design ใน Bootstrap

Grid System ใน Bootstrap เป็นระบบเลย์เอาต์แบบยืดหยุ่นที่ช่วยให้การจัดวางองค์ประกอบต่างๆ ในหน้าเว็บเป็นระเบียบและสอดคล้องกับทุกขนาดหน้าจอ โดยใช้หลักการแบ่งพื้นที่ออกเป็น 12 คอลัมน์ในแต่ละแถว (Row) นักพัฒนาสามารถควบคุมขนาดและการจัดวางของคอลัมน์ได้อย่างยืดหยุ่นผ่านคลาส เช่น .col-*, .col-md-*, .col-lg-* โดยใช้งานร่วมกับ .row เพื่อกำหนดแถว และใช้ Breakpoints เพื่อให้เลย์เอาต์ปรับเปลี่ยนได้ตามขนาดอุปกรณ์ เช่น มือถือ แท็บเล็ต หรือเดสก์ท็อป นอกจากนี้ Bootstrap ยังมีคลาส .container สำหรับกำหนดความกว้างคงที่ และ .container-fluid สำหรับความกว้างเต็มหน้าจอ เพื่อช่วยควบคุมขอบเขตของเนื้อหาได้อย่างเหมาะสม

ในด้าน Layout โดยรวม Bootstrap ช่วยให้นักพัฒนาสามารถวางโครงสร้างของเว็บไซต์ได้อย่างเป็นระบบ โดยสามารถผสมผสาน Grid กับองค์ประกอบต่างๆ เช่น Header, Sidebar, Content และ Footer ได้อย่างง่ายดาย ทำให้สามารถสร้างเลย์เอาต์ที่ดูดีและใช้งานได้จริงในทุกบริบทการออกแบบ ไม่ว่าจะเป็นหน้า Landing Page, Dashboard หรือ Blog รวมถึงสามารถปรับแต่งแต่ละส่วนให้ตอบสนองการใช้งานในอุปกรณ์ต่างๆ ได้อย่างลงตัว ส่งผลให้การพัฒนาเว็บมีประสิทธิภาพมากขึ้นและใช้เวลาน้อยลง.

Grid System (ระบบ 12 คอลัมน์) ใน Bootstrap

☐ **Grid System คืออะไร?**

Grid System ใน Bootstrap เป็น **ระบบเลย์เอาต์แบบยืดหยุ่น** ที่ช่วยให้สามารถจัดวางองค์ประกอบต่าง ๆ ในหน้าเว็บได้ง่ายขึ้น โดยใช้หลักการแบ่งพื้นที่เป็น **12 คอลัมน์** (Columns) ในแต่ละแถว (Row)

Bootstrap Grid ใช้ **Flexbox** เป็นพื้นฐาน จึงสามารถปรับขนาดอัตโนมัติตามขนาดหน้าจอ (Responsive) และรองรับการออกแบบเว็บแบบ Mobile-First

---

☐ **หน้าที่ของ Grid System**

1. **จัดระเบียบเลย์เอาต์ของเว็บ** – ทำให้การจัดวางองค์ประกอบในหน้าเว็บเป็นระเบียบมากขึ้น
2. **รองรับ Responsive Design** – ปรับขนาดขององค์ประกอบให้เหมาะกับทุกขนาดหน้าจอ
3. **ลดภาระการเขียน CSS** – ใช้คลาสสำเร็จรูปในการกำหนดเลย์เอาต์แทนการเขียน CSS เอง
4. **ปรับแต่งขนาดของคอลัมน์ได้ง่าย** – สามารถกำหนดความกว้างของแต่ละคอลัมน์ได้ตามต้องการ

☐ **โครงสร้างของ Grid System**

Bootstrap Grid System ประกอบด้วย 3 ส่วนหลัก

1. **Container** – เป็นตัวห่อหุ้ม Grid ทั้งหมด
2. **Row** – ใช้แบ่งแถว (Row) ภายใน Container
3. **Columns** – ใช้แบ่งคอลัมน์ภายในแต่ละ Row โดยมี 12 คอลัมน์เป็นพื้นฐาน

☐ **ตัวอย่างการใช้งาน Grid System**

☐ **ตัวอย่าง 1: แบ่งคอลัมน์เท่า ๆ กัน**

หน้าเว็บแบ่งเป็น 3 ส่วนที่มีขนาดเท่ากัน (แต่ละคอลัมน์ใช้พื้นที่ 4/12 ของหน้าจอ)

☐ **ตัวอย่าง 2: แบ่งคอลัมน์ไม่เท่ากัน**

สามารถกำหนดให้บางคอลัมน์ใหญ่กว่าหรือเล็กกว่าได้ เช่น 8 ส่วน + 4 ส่วน

☐ **ตัวอย่าง 3: Responsive Grid**

กำหนดให้แต่ละคอลัมน์แสดงผลแตกต่างกันตามขนาดหน้าจอ เช่น

- col-md-6 หมายถึง **เมื่อหน้าจอขนาดกลางขึ้นไป** คอลัมน์นี้จะใช้พื้นที่ **6/12** ของหน้าจอ
- col-sm-12 col-md-6 หมายถึง **หน้าจอเล็กใช้เต็มแถว (12/12) แต่หน้าจอกลางขึ้นไปใช้ครึ่งแถว (6/12)**

☐ **ข้อดีของ Bootstrap Grid System**

☐ ช่วยให้การพัฒนาเว็บเร็วขึ้น

☐ รองรับการปรับแต่งเลย์เอาต์แบบ Responsive

☐ ใช้งานง่ายโดยไม่ต้องเขียน CSS เอง

☐ ปรับแต่งขนาดคอลัมน์ได้อย่างยืดหยุ่น

☐ **สรุป**

Bootstrap Grid System เป็นระบบแบ่งคอลัมน์ 12 ช่อง ที่ช่วยให้นักพัฒนาจัดวางเลย์เอาต์ของหน้าเว็บได้ง่าย รองรับการทำงานแบบ Responsive และสามารถกำหนดขนาดของแต่ละคอลัมน์ได้ตามต้องการ ทำให้การออกแบบหน้าเว็บเป็นระเบียบและยืดหยุ่นมากขึ้น ☐

ข้อมูลเพิ่มเติมเกี่ยวกับ Bootstrap Grid System

☐ **1. โครงสร้างหลักของ Bootstrap Grid System**

Bootstrap Grid ใช้ **Flexbox** เป็นพื้นฐาน และมีโครงสร้าง 3 ระดับหลัก:

1. **Container** – ใช้กำหนดขอบเขตของ Grid (เช่น .container, .container-fluid)
2. **Row** – ใช้จัดกลุ่มคอลัมน์เข้าด้วยกัน (.row)
3. **Column** – ใช้แบ่งคอลัมน์ภายใน Row (.col-*)

☐ **2. ประเภทของ Container ใน Grid System**

ประเภท	คุณสมบัติ
.container	มีขนาดคงที่ ปรับตามขนาดหน้าจอ
.container-fluid	กว้างเต็มจอเสมอ

☐ **ข้อดีของ .container-fluid** คือเหมาะสำหรับเว็บที่ต้องการใช้พื้นที่เต็มหน้าจอทุกขนาด

☐ **3. คอลัมน์ใน Grid System**

Bootstrap Grid ใช้หลักการแบ่ง **12 คอลัมน์ต่อแถว** เช่น

- col-6 หมายถึง **ครึ่งหนึ่งของแถว (6/12 หรือ 50%)**
- col-4 หมายถึง **1/3 ของแถว (4/12 หรือ 33.33%)**
- col-3 หมายถึง **1/4 ของแถว (3/12 หรือ 25%)**

☐ **ข้อสังเกต:**

- หากคอลัมน์รวมกันเกิน 12 คอลัมน์ จะขึ้นบรรทัดใหม่อัตโนมัติ
- ถ้าไม่กำหนดขนาด (col เฉยๆ) Bootstrap จะแบ่งคอลัมน์ให้เท่ากัน

☐ **4. Breakpoints (จุดเปลี่ยนขนาดจอ) ของ Bootstrap Grid**

Bootstrap รองรับการแสดงผลที่แตกต่างกันในแต่ละขนาดหน้าจอ

คำสั่ง	ขนาดหน้าจอ	ความกว้าง
col-	Extra small (XS)	น้อยกว่า 576px
col-sm-	Small (SM)	576px ขึ้นไป
col-md-	Medium (MD)	768px ขึ้นไป
col-lg-	Large (LG)	992px ขึ้นไป
col-xl-	Extra Large (XL)	1200px ขึ้นไป
col-xxl-	Extra Extra Large (XXL)	1400px ขึ้นไป

☐ **ตัวอย่างการใช้งาน**

- col-md-6 หมายถึง **เมื่อหน้าจอขนาดกลาง (≥768px) คอลัมน์นี้จะใช้ 6/12 ของพื้นที่**
- col-sm-12 col-md-6 หมายถึง **หน้าจอเล็ก (XS, SM) ใช้เต็มแถว แต่หน้าจอกลางขึ้นไปใช้ครึ่งแถว**

☐ **5. การจัดตำแหน่งคอลัมน์ (Alignment & Ordering)**

☐ **การจัดเรียงแนวตั้ง (Vertical Alignment)**

ใช้ **.align-items-*** หรือ **.align-self-*** เช่น

- .align-items-start (ชิดด้านบน)
- .align-items-center (กึ่งกลาง)
- .align-items-end (ชิดด้านล่าง)

☐ **การจัดเรียงแนวนอน (Horizontal Alignment)**

ใช้ **.justify-content-*** เช่น

- .justify-content-start (ชิดซ้าย)
- .justify-content-center (กึ่งกลาง)
- .justify-content-end (ชิดขวา)
- .justify-content-between (กระจายระยะห่าง)

☐ **การเรียงลำดับคอลัมน์ (Column Ordering)**

สามารถใช้ .order-* เพื่อกำหนดลำดับของคอลัมน์ เช่น

- .order-1 – ให้คอลัมน์แสดงเป็นลำดับที่ 1
- .order-2 – ให้คอลัมน์แสดงเป็นลำดับที่ 2

☐ **6. การซ้อนคอลัมน์ (Nesting Columns)**

สามารถซ้อนคอลัมน์ภายในคอลัมน์ได้โดยใช้ **Row ใหม่ภายในคอลัมน์** เช่น

- คอลัมน์หลัก (col-6)

- ภายในมีคอลัมน์ย่อย (col-6 และ col-6 อีกชุด)

☐ **ข้อดีของ Nesting** คือสามารถสร้างเลย์เอาต์ที่ซับซ้อนขึ้นได้

☐ **7. ตัวอย่างการใช้งานจริงของ Grid System**

☐ **ตัวอย่างการแบ่งเลย์เอาต์ของหน้าเว็บ:**

1. **เว็บบล็อก**
   - col-12 (ส่วนหัว)
   - col-md-8 (เนื้อหาหลัก)
   - col-md-4 (Sidebar)
2. **เว็บไซต์ร้านค้าออนไลน์**
   - col-lg-3 (เมนูสินค้า)
   - col-lg-6 (สินค้าแนะนำ)
   - col-lg-3 (ตะกร้าสินค้า)
3. **Dashboard (แสดงข้อมูล)**
   - col-4 (สถิติที่ 1)
   - col-4 (สถิติที่ 2)
   - col-4 (สถิติที่ 3)

☐ **8. สรุปข้อดีของ Bootstrap Grid System**

☐ **ใช้งานง่าย** – มีคลาสสำเร็จรูปให้ใช้ ไม่ต้องเขียน CSS เอง

☐ **รองรับ Responsive** – ปรับขนาดคอลัมน์อัตโนมัติตามขนาดจอ

☐ **ยืดหยุ่นสูง** – ปรับแต่งขนาดและตำแหน่งของคอลัมน์ได้อย่างอิสระ

☐ **ใช้ได้ทุกขนาดหน้าจอ** – มี Breakpoints ให้เลือกใช้ตามขนาดจอ

☐ **สรุปโดยย่อ**

Grid System ใน Bootstrap คือ **ระบบแบ่ง 12 คอลัมน์** ที่ช่วยให้การจัดเลย์เอาต์ของหน้าเว็บง่ายขึ้น สามารถปรับแต่งขนาดของคอลัมน์ให้เหมาะกับแต่ละขนาดหน้าจอ และรองรับการออกแบบแบบ Responsive อย่างมีประสิทธิภาพ

ตัวอย่างโค้ด Bootstrap Grid System

โค้ดตัวอย่างนี้จะแสดงการใช้งาน **Grid System ของ Bootstrap** รวมถึง **Container, Row, Column, Responsive Breakpoints และการจัดเรียงคอลัมน์**

☐ **โครงสร้างของตัวอย่างนี้**

☐ แสดงการใช้งาน **Container, Row และ Column**

☐ มีการใช้ **Responsive Breakpoints (col-sm-*, col-md-*, col-lg-*)**

☐ ใช้ **การจัดตำแหน่ง (justify-content-center, align-items-center)**

☐ ใช้ **การจัดลำดับ (order-*)**

☐ **โค้ดตัวอย่าง Bootstrap Grid System**

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap Grid System Example</title>
    <!-- เรียกใช้งาน Bootstrap ผ่าน CDN -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
    <style>
        .box {
            background-color: lightblue;
            text-align: center;
            padding: 20px;
            border: 1px solid #ddd;
        }
    </style>
</head>
<body>

    <!-- Container ใช้กำหนดขอบเขตของ Grid -->
    <div class="container mt-4">
        <h2 class="text-center">ตัวอย่างการใช้งาน Bootstrap Grid System</h2>

        <!-- แถวที่ 1: แบ่งคอลัมน์เท่ากัน (3 ส่วน) -->
        <div class="row">
            <div class="col-md-4 box">Column 1</div>
```

```
        <div class="col-md-4 box">Column 2</div>
        <div class="col-md-4 box">Column 3</div>
    </div>

    <!-- แถวที่ 2: คอลัมน์ขนาดไม่เท่ากัน -->
    <div class="row mt-3">
        <div class="col-md-8 box">Column 8/12</div>
        <div class="col-md-4 box">Column 4/12</div>
    </div>

    <!-- แถวที่ 3: คอลัมน์แบบ Responsive -->
    <div class="row mt-3">
        <div class="col-sm-12 col-md-6 col-lg-3 box">Column 1 (ปรับขนาดตามจอ)</div>
        <div class="col-sm-12 col-md-6 col-lg-3 box">Column 2 (ปรับขนาดตามจอ)</div>
        <div class="col-sm-12 col-md-6 col-lg-3 box">Column 3 (ปรับขนาดตามจอ)</div>
        <div class="col-sm-12 col-md-6 col-lg-3 box">Column 4 (ปรับขนาดตามจอ)</div>
    </div>

    <!-- แถวที่ 4: การจัดตำแหน่งแนวนอน -->
    <div class="row mt-3 justify-content-center">
        <div class="col-md-4 box">Centered Column</div>
    </div>

    <!-- แถวที่ 5: การจัดตำแหน่งแนวตั้ง -->
    <div class="row mt-3 align-items-center" style="height: 150px; background-color:
#f8f9fa;">
        <div class="col-md-4 box">Column 1</div>
        <div class="col-md-4 box">Column 2 (กึ่งกลาง)</div>
        <div class="col-md-4 box">Column 3</div>
    </div>

    <!-- แถวที่ 6: การเรียงลำดับคอลัมน์ -->
    <div class="row mt-3">
        <div class="col-md-4 order-md-2 box">Column 1 (แสดงเป็นอันดับ 2)</div>
```

```
            <div class="col-md-4 order-md-1 box">Column 2 (แสดงเป็นอันดับ 1)</div>
            <div class="col-md-4 order-md-3 box">Column 3 (แสดงเป็นอันดับ 3)</div>
        </div>
    </div>

    <!-- เรียกใช้ Bootstrap JavaScript -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

□ คำอธิบายโค้ด

1. **กำหนด Container (.container)**
   - ใช้ **container** เพื่อให้เนื้อหาอยู่กึ่งกลางของหน้าจอ
2. **กำหนดแถว (.row) และคอลัมน์ (.col-*)**
   - **แถวที่ 1**: คอลัมน์แบ่งเป็น 3 ส่วนเท่ากัน (col-md-4)
   - **แถวที่ 2**: คอลัมน์แบ่งเป็น 8/12 และ 4/12
   - **แถวที่ 3**: คอลัมน์ที่รองรับ **Responsive Design** (col-sm-12 col-md-6 col-lg-3)
   - **แถวที่ 4**: ใช้ **.justify-content-center** เพื่อจัดคอลัมน์ตรงกลาง
   - **แถวที่ 5**: ใช้ **.align-items-center** เพื่อจัดตำแหน่งแนวตั้ง
   - **แถวที่ 6**: ใช้ **.order-md-*** เพื่อกำหนดลำดับการแสดงผลของคอลัมน์

□ ผลลัพธ์ที่ได้

- **แถวที่ 1:** คอลัมน์ 3 ส่วนเท่ากัน
- **แถวที่ 2:** คอลัมน์ 8 ส่วน + 4 ส่วน
- **แถวที่ 3:** คอลัมน์ปรับขนาดตามหน้าจอ (Responsive)
- **แถวที่ 4:** คอลัมน์จัดให้อยู่กึ่งกลาง
- **แถวที่ 5:** คอลัมน์จัดตำแหน่งแนวตั้งให้อยู่กึ่งกลาง
- **แถวที่ 6:** การเรียงลำดับของคอลัมน์เปลี่ยนไปตามที่กำหนด

□ สรุป

- **Bootstrap Grid System** ใช้ **Container, Row** และ **Column** ในการแบ่งโครงสร้างเว็บ
- รองรับ **Responsive Design** ด้วย **Breakpoint (col-sm-*, col-md-*, col-lg-*)**

- **มีการจัดตำแหน่งคอลัมน์** (แนวนอนและแนวตั้ง)
- **สามารถปรับแต่งการเรียงลำดับคอลัมน์** (order-md-*)

ตัวอย่างเพิ่มเติมอีก **5 ตัวอย่าง** ที่แสดงการใช้งาน **Bootstrap Grid System** ในสถานการณ์ต่างๆ

ตัวอย่างที่ 1: Layout แบบ Blog (เนื้อหาหลัก + Sidebar)

โครงสร้าง:

- เนื้อหาหลัก (col-md-8)
- Sidebar (col-md-4)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-md-8 box">เนื้อหาหลัก (Main Content)</div>
        <div class="col-md-4 box">Sidebar</div>
    </div>
</div>
```

ตัวอย่างที่ 2: Layout แบบ Header + 3 Column Content

โครงสร้าง:

- Header (col-12)
- คอลัมน์เนื้อหา 3 ส่วน (col-md-4)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-12 box">ส่วนหัวเว็บไซต์ (Header)</div>
    </div>
    <div class="row mt-3">
        <div class="col-md-4 box">คอลัมน์ที่ 1</div>
        <div class="col-md-4 box">คอลัมน์ที่ 2</div>
        <div class="col-md-4 box">คอลัมน์ที่ 3</div>
    </div>
</div>
```

ตัวอย่างที่ 3: Layout แบบ Dashboard (Grid 4 คอลัมน์)

โครงสร้าง:

- 4 คอลัมน์ (col-md-3) รองรับหน้าจอขนาดเล็ก

```
<div class="container mt-4">
    <div class="row">
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 1</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 2</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 3</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 4</div>
    </div>
</div>
```

□ **ตัวอย่างที่ 4: Layout แบบ 2 คอลัมน์ พร้อมจัดตำแหน่ง**

□ **โครงสร้าง:**

- 2 คอลัมน์ (col-md-6)
- จัดตำแหน่งตรงกลาง (justify-content-center)

```
<div class="container mt-4">
    <div class="row justify-content-center">
        <div class="col-md-6 box">□ เนื้อหาซ้าย</div>
        <div class="col-md-6 box">□ เนื้อหาขวา</div>
    </div>
</div>
```

□ **ตัวอย่างที่ 5: Layout แบบ Nested Grid (Grid ซ้อนกัน)**

□ **โครงสร้าง:**

- คอลัมน์หลัก (col-md-6)
- ภายในมีคอลัมน์ย่อย (col-6)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-md-6 box">
            □ คอลัมน์หลัก (50%)
            <div class="row mt-3">
                <div class="col-6 box">□ คอลัมน์ย่อย 1</div>
                <div class="col-6 box">□ คอลัมน์ย่อย 2</div>
            </div>
        </div>
        <div class="col-md-6 box">□ คอลัมน์หลัก (50%)</div>
```

```
    </div>
</div>
```

□ **สรุป**

- ตัวอย่าง **#1**: Layout แบบ Blog (เนื้อหาหลัก + Sidebar)
- ตัวอย่าง **#2**: Layout แบบ Header + 3 Column Content
- ตัวอย่าง **#3**: Layout แบบ Dashboard (Grid 4 คอลัมน์)
- ตัวอย่าง **#4**: Layout แบบ 2 คอลัมน์ พร้อมจัดตำแหน่ง
- ตัวอย่าง **#5**: Layout แบบ Nested Grid (Grid ซ้อนกัน)

ตัวอย่างเพิ่มเติมอีก **5 ตัวอย่าง** ที่แสดงการใช้งาน **Bootstrap Grid System** ในรูปแบบต่างๆ

□ **ตัวอย่างที่ 6: Layout แบบ Gallery (รูปภาพหลายช่อง)**

□ **โครงสร้าง:**

- ใช้ **Grid 3 คอลัมน์** (col-md-4)
- รองรับหน้าจอขนาดเล็ก (col-sm-6)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 1</div>
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 2</div>
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 3</div>
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 4</div>
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 5</div>
        <div class="col-sm-6 col-md-4 box">□ รูปภาพ 6</div>
    </div>
</div>
```

□ **ตัวอย่างที่ 7: Layout แบบ Pricing Table (แสดงแผนราคา)**

□ **โครงสร้าง:**

- คอลัมน์ 3 ส่วน (col-md-4)
- ปรับขนาดตามจอ (col-sm-6, col-lg-3)

```
<div class="container mt-4">
    <div class="row">
```

```
        <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan A</div>
        <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan B</div>
        <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan C</div>
        <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan D</div>
    </div>
</div>
```

□ ตัวอย่างที่ 8: Layout แบบ Contact Form (ฟอร์มติดต่อ)

□ โครงสร้าง:

- ฟอร์มแบบ **2 คอลัมน์** (col-md-6)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-md-6 box">□ ฟอร์มติดต่อ</div>
        <div class="col-md-6 box">□ ที่อยู่สำนักงาน</div>
    </div>
</div>
```

□ ตัวอย่างที่ 9: Layout แบบ Hero Section (แสดงข้อความเด่น)

□ โครงสร้าง:

- ข้อความอยู่ตรงกลางจอ (justify-content-center, align-items-center)

```
<div class="container mt-4">
    <div class="row justify-content-center align-items-center" style="height: 300px; background-color: #f8f9fa;">
        <div class="col-md-6 text-center box">□ ยินดีต้อนรับสู่เว็บไซต์ของเรา</div>
    </div>
</div>
```

□ ตัวอย่างที่ 10: Layout แบบ Footer (ส่วนท้ายเว็บไซต์)

□ โครงสร้าง:

- Footer แบ่งเป็น 3 ส่วน (col-md-4)
- ปรับขนาดอัตโนมัติ (col-sm-12)

```
<div class="container mt-4">
    <div class="row">
        <div class="col-sm-12 col-md-4 box">□ About Us</div>
```

```
        <div class="col-sm-12 col-md-4 box">□ Quick Links</div>
        <div class="col-sm-12 col-md-4 box">□ Contact</div>
    </div>
</div>
```

□ สรุป

- ตัวอย่าง **#6**: Layout แบบ Gallery (รูปภาพหลายช่อง)
- ตัวอย่าง **#7**: Layout แบบ Pricing Table (แสดงแผนราคา)
- ตัวอย่าง **#8**: Layout แบบ Contact Form (ฟอร์มติดต่อ)
- ตัวอย่าง **#9**: Layout แบบ Hero Section (แสดงข้อความเด่น)
- ตัวอย่าง **#10**: Layout แบบ Footer (ส่วนท้ายเว็บไซต์)

ตัวอย่าง **Bootstrap Grid System** ที่ครบถ้วน รวมทั้ง 10 ตัวอย่างที่กล่าวถึงก่อนหน้านี้

□ โค้ด HTML เต็ม

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>ตัวอย่าง Bootstrap Grid</title>
    <!-- Bootstrap 5 CDN -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">
    <style>
        .box {
            background: #f8f9fa;
            padding: 20px;
            border: 1px solid #ddd;
            text-align: center;
            margin-bottom: 10px;
            font-weight: bold;
        }
    </style>
```

```
</head>
<body>

<div class="container mt-4">
    <!-- ตัวอย่าง 1: Layout Blog -->
    <div class="row">
        <div class="col-md-8 box">□ เนื้อหาหลัก (Main Content)</div>
        <div class="col-md-4 box">□ Sidebar</div>
    </div>

    <!-- ตัวอย่าง 2: Header + 3 Column Content -->
    <div class="row">
        <div class="col-12 box">□ ส่วนหัวเว็บไซต์ (Header)</div>
    </div>
    <div class="row">
        <div class="col-md-4 box">□ คอลัมน์ที่ 1</div>
        <div class="col-md-4 box">□ คอลัมน์ที่ 2</div>
        <div class="col-md-4 box">□ คอลัมน์ที่ 3</div>
    </div>

    <!-- ตัวอย่าง 3: Dashboard 4 คอลัมน์ -->
    <div class="row">
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 1</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 2</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 3</div>
        <div class="col-sm-6 col-md-3 box">□ การ์ดที่ 4</div>
    </div>

    <!-- ตัวอย่าง 4: Layout 2 คอลัมน์ -->
    <div class="row justify-content-center">
        <div class="col-md-6 box">□ เนื้อหาซ้าย</div>
        <div class="col-md-6 box">□ เนื้อหาขวา</div>
    </div>
```

```
<!-- ตัวอย่าง 5: Nested Grid -->
<div class="row">
    <div class="col-md-6 box">
        □ คอลัมน์หลัก (50%)
        <div class="row">
            <div class="col-6 box">□ คอลัมน์ย่อย 1</div>
            <div class="col-6 box">□ คอลัมน์ย่อย 2</div>
        </div>
    </div>
    <div class="col-md-6 box">□ คอลัมน์หลัก (50%)</div>
</div>

<!-- ตัวอย่าง 6: Gallery -->
<div class="row">
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 1</div>
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 2</div>
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 3</div>
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 4</div>
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 5</div>
    <div class="col-sm-6 col-md-4 box">□ รูปภาพ 6</div>
</div>

<!-- ตัวอย่าง 7: Pricing Table -->
<div class="row">
    <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan A</div>
    <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan B</div>
    <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan C</div>
    <div class="col-sm-6 col-md-4 col-lg-3 box">□ Plan D</div>
</div>

<!-- ตัวอย่าง 8: Contact Form -->
<div class="row">
    <div class="col-md-6 box">□ ฟอร์มติดต่อ</div>
    <div class="col-md-6 box">□ ที่อยู่สำนักงาน</div>
```

```
    </div>

    <!-- ตัวอย่าง 9: Hero Section -->
    <div class="row justify-content-center align-items-center" style="height: 300px; background-color: #f8f9fa;">
        <div class="col-md-6 text-center box">□ ยินดีต้อนรับสู่เว็บไซต์ของเรา</div>
    </div>

    <!-- ตัวอย่าง 10: Footer -->
    <div class="row">
        <div class="col-sm-12 col-md-4 box">□ About Us</div>
        <div class="col-sm-12 col-md-4 box">□ Quick Links</div>
        <div class="col-sm-12 col-md-4 box">□ Contact</div>
    </div>
</div>

<!-- Bootstrap 5 JavaScript -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>

</body>
</html>
```

□ **คำอธิบาย**

□ **Grid System** ของ Bootstrap แบ่งโครงสร้างออกเป็น **12 คอลัมน์** และรองรับ **Responsive Design**

□ ใช้ **คลาส .col-*** เพื่อตั้งค่าขนาดคอลัมน์ที่แตกต่างกันตามหน้าจอ

□ ใช้ **CSS .box** เพื่อเพิ่มพื้นหลังและเส้นขอบให้ดูชัดเจน

□ ใช้ **Bootstrap 5 CDN** ทำให้รันได้ทันที **ไม่ต้องติดตั้งแยก**

□ **สรุป**

- □ ตัวอย่าง **#1 - #10** ครอบคลุมการใช้งาน **Grid System** ในรูปแบบต่างๆ
- □ สามารถ **คัดลอกโค้ดนี้ไปใช้งานได้เลย**
- □ รองรับทั้ง **Desktop / Tablet / Mobile**

Container ใน Bootstrap คืออะไร?

Container เป็นองค์ประกอบหลักของ **Bootstrap Grid System** ที่ใช้กำหนดความกว้างของเนื้อหา โดยมี **2 ประเภทหลัก** ได้แก่

1. **.container** ⟶ กำหนดขนาดความกว้างตาม **breakpoints** (กำหนดขนาดอัตโนมัติตามหน้าจอ)
2. **.container-fluid** ⟶ ใช้ความกว้างเต็มหน้าจอ (100% ของ viewport)

---

1. .container (Fixed Width)

- ขนาดความกว้างของ .container จะเปลี่ยนไปตาม **breakpoints** ของ Bootstrap
- ใช้เมื่อต้องการให้เนื้อหาอยู่กึ่งกลางและมี **ขนาดที่กำหนดไว้**

ตัวอย่างการใช้งาน

```
<div class="container">
    <h3> ตัวอย่าง Container</h3>
    <p>Container นี้จะปรับขนาดตาม breakpoint ของ Bootstrap</p>
</div>
```

ขนาดความกว้างของ .container ตามหน้าจอ

Breakpoint	ความกว้าง .container
Extra Small (xs) (<576px)	100%
Small (sm) (≥576px)	540px
Medium (md) (≥768px)	720px
Large (lg) (≥992px)	960px
Extra Large (xl) (≥1200px)	1140px
XXL (xxl) (≥1400px)	1320px

---

2. .container-fluid (Full Width)

- ใช้เต็ม **100% ความกว้างของหน้าจอ**
- เหมาะสำหรับ **Landing Pages, Dashboard, และ Layout ที่ต้องการเต็มจอ**

ตัวอย่างการใช้งาน

```
<div class="container-fluid">
```

```
    <h3>□ ตัวอย่าง Container Fluid</h3>
    <p>Container นี้จะมีความกว้างเต็มหน้าจอเสมอ</p>
</div>
```

□ 3. .container-{breakpoint} (กำหนดขนาดที่แน่นอน)

นอกจาก .container และ .container-fluid แล้ว **Bootstrap 5+** ยังรองรับ .container-{breakpoint} เพื่อกำหนดความกว้างแบบเฉพาะเจาะจง

□ ตัวอย่างการใช้งาน

```
<div class="container-md">
    <h3>□ ตัวอย่าง Container-MD</h3>
    <p>Container นี้จะเป็น Fluid ในหน้าจอขนาดเล็ก และเป็น Fixed เมื่อขนาดมากกว่า md</p>
</div>
```

□ container-sm ⟶ Fluid จนถึง sm จากนั้นจะเป็น Fixed

□ container-md ⟶ Fluid จนถึง md จากนั้นจะเป็น Fixed

□ container-lg ⟶ Fluid จนถึง lg จากนั้นจะเป็น Fixed

□ เปรียบเทียบการใช้งาน

```
<div class="container bg-light p-3">
    <h3>□ .container</h3>
    <p>มีความกว้างคงที่และอยู่ตรงกลาง</p>
</div>

<div class="container-fluid bg-warning p-3">
    <h3>□ .container-fluid</h3>
    <p>ขยายเต็มความกว้างของหน้าจอ</p>
</div>

<div class="container-lg bg-primary text-white p-3">
    <h3>□ .container-lg</h3>
    <p>จะเป็น Fluid จนถึง lg ขึ้นไป</p>
</div>
```

□ สรุป

ประเภท	ลักษณะการทำงาน
.container	กำหนดขนาดความกว้างตาม **breakpoints**
.container-fluid	ขยายเต็มความกว้างของหน้าจอ **(100%)**
.container-{breakpoint}	กำหนดเป็น Fluid จนถึงขนาดที่กำหนด

□ **ใช้ .container** ⟶ เมื่ออยากให้เนื้อหาตรงกลางและปรับตามขนาดหน้าจอ

□ **ใช้ .container-fluid** ⟶ เมื่ออยากให้เนื้อหาขยายเต็มจอ

□ **ใช้ .container-{breakpoint}** ⟶ เมื่ออยากให้เนื้อหา Fluid เฉพาะบางขนาด

□ ข้อมูลเพิ่มเติมเกี่ยวกับ Bootstrap Container

1□ .container และ .container-fluid ใช้งานร่วมกับ Bootstrap Grid System

Bootstrap Container มักใช้เป็นโครงร่างหลักของหน้าเว็บ และทำงานร่วมกับ **Grid System (row และ col-*)** เพื่อช่วยจัดวางเลย์เอาต์

□ **ตัวอย่างโครงสร้างที่ใช้ร่วมกับ Bootstrap Grid System**

```
<div class="container">
    <div class="row">
        <div class="col-md-6 bg-light p-3">□ คอลัมน์ซ้าย</div>
        <div class="col-md-6 bg-secondary text-white p-3">□ คอลัมน์ขวา</div>
    </div>
</div>
```

□ .container ใช้ควบคุมขนาดของเนื้อหา

□ .row ใช้จัดระเบียบโครงสร้าง

□ .col-md-6 ใช้แบ่ง 50% ของพื้นที่

2□ การใช้ .container-fluid กับ Grid System

container-fluid เหมาะสำหรับ **Dashboard** หรือ **Landing Page** ที่ต้องการใช้ความกว้างเต็มจอ

□ **ตัวอย่างการใช้ .container-fluid กับ Grid System**

```
<div class="container-fluid">
    <div class="row">
        <div class="col-md-4 bg-primary text-white p-3">□ คอลัมน์ 1</div>
        <div class="col-md-4 bg-success text-white p-3">□ คอลัมน์ 2</div>
        <div class="col-md-4 bg-danger text-white p-3">□ คอลัมน์ 3</div>
```

```
    </div>
</div>
```

□ .container-fluid ใช้เต็มหน้าจอ

□ .col-md-4 แบ่งคอลัมน์เท่าๆ กัน

3□ การใช้ .container-{breakpoint} กับ Responsive Design

Bootstrap 5+ รองรับ .container-{breakpoint} เพื่อให้ Fluid จนถึงขนาดที่กำหนด แล้วเปลี่ยนเป็น Fixed

□ **ตัวอย่างการใช้ .container-md**

```
<div class="container-md bg-info text-white p-3">
    <h3>□ .container-md</h3>
    <p>เต็มหน้าจอในขนาดเล็ก แต่จะเป็น Fixed เมื่อหน้าจอใหญ่ขึ้น</p>
</div>
```

□ .container-md ⟶ เต็มจอเมื่อขนาด < md (768px) และจะเป็น Fixed เมื่อ ≥768px

□ **การทำงานของ .container-{breakpoint}**

คลาสที่ใช้	Full Width จนถึง	เปลี่ยนเป็น Fixed เมื่อ
.container-sm	<576px	≥576px
.container-md	<768px	≥768px
.container-lg	<992px	≥992px
.container-xl	<1200px	≥1200px
.container-xxl	<1400px	≥1400px

4□ ตัวอย่างการใช้งาน .container และ .container-fluid ในเว็บไซต์จริง

□ **ตัวอย่าง 1: โครงสร้างเว็บไซต์ทั่วไป**

```
<div class="container">
    <header class="bg-primary text-white text-center p-3">□ Header</header>

    <div class="row">
        <aside class="col-md-3 bg-light p-3">□ เมนู</aside>
        <main class="col-md-9 p-3">□ เนื้อหาหลัก</main>
    </div>
```

```
    <footer class="bg-dark text-white text-center p-3">□ Footer</footer>
</div>
```

□ ใช้ .container เพื่อกำหนดโครงสร้างเว็บไซต์

□ ใช้ .row และ .col-* แบ่งโซน **Header, Sidebar, Content, Footer**

□ **ตัวอย่าง 2: Layout แบบเต็มจอด้วย .container-fluid**

```
<div class="container-fluid">
    <header class="bg-dark text-white text-center p-3">□ Header</header>

    <div class="row">
        <div class="col-md-4 bg-warning p-3">□ คอลัมน์ซ้าย</div>
        <div class="col-md-4 bg-info text-white p-3">□ คอลัมน์กลาง</div>
        <div class="col-md-4 bg-success text-white p-3">□ คอลัมน์ขวา</div>
    </div>

    <footer class="bg-secondary text-white text-center p-3">□ Footer</footer>
</div>
```

□ ใช้ .container-fluid ทำให้เนื้อหาเต็มจอ

□ ใช้ **สีพื้นหลัง** แสดงผลคอลัมน์ชัดเจน

□ **สรุป**

ประเภท	ใช้เมื่อไหร่?	ลักษณะการทำงาน
.container	ใช้เมื่อ **ต้องการให้เนื้อหาอยู่ตรงกลาง** และมีขนาดที่กำหนด	ความกว้าง **เปลี่ยนแปลงตาม breakpoint**
.container-fluid	ใช้เมื่อ **ต้องการเต็มหน้าจอ** (100% width)	ความกว้าง **เต็มจอตลอดเวลา**
.container-{breakpoint}	ใช้เมื่อ **ต้องการให้ Fluid จนถึงขนาดที่กำหนด**	เปลี่ยนจาก **Fluid ⟶ Fixed** ตาม breakpoint

□ **.container** ⟶ ใช้สำหรับ **เว็บปกติ**

□ **.container-fluid** ⟶ ใช้สำหรับ **Dashboard / Landing Page**

□ **.container-{breakpoint}** ⟶ ใช้สำหรับ **Responsive Layout**

ตัวอย่างโค้ด HTML เต็มรูปแบบที่แสดงการใช้ .container, .container-fluid และ .container-{breakpoint}

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ตัวอย่าง Container ใน Bootstrap</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css">
</head>
<body>

    <!-- ตัวอย่าง 1: ใช้ .container (Fixed Width) -->
    <div class="container mt-4">
        <h2 class="text-center">□ ตัวอย่าง .container</h2>
        <p class="text-center">เนื้อหาจะอยู่ตรงกลางและขยายตาม breakpoint</p>
        <div class="row">
            <div class="col-md-6 bg-light p-3 border">□ คอลัมน์ 1</div>
            <div class="col-md-6 bg-secondary text-white p-3 border">□ คอลัมน์ 2</div>
        </div>
    </div>

    <!-- ตัวอย่าง 2: ใช้ .container-fluid (Full Width) -->
    <div class="container-fluid bg-warning text-dark mt-4 p-4">
        <h2 class="text-center">□ ตัวอย่าง .container-fluid</h2>
        <p class="text-center">เนื้อหานี้จะเต็มจอ 100% ตลอดเวลา</p>
        <div class="row">
            <div class="col-md-4 bg-primary text-white p-3">□ คอลัมน์ 1</div>
            <div class="col-md-4 bg-success text-white p-3">□ คอลัมน์ 2</div>
            <div class="col-md-4 bg-danger text-white p-3">□ คอลัมน์ 3</div>
        </div>
    </div>
```

```
<!-- ตัวอย่าง 3: ใช้ .container-md (Responsive Container) -->
<div class="container-md bg-info text-white mt-4 p-4">
    <h2 class="text-center">□ ตัวอย่าง .container-md</h2>
    <p class="text-center">ขนาดเล็กจะเต็มจอ แต่จะเป็น Fixed เมื่อหน้าจอใหญ่ขึ้น</p>
    <div class="row">
        <div class="col-md-3 bg-dark p-3">□ เมนู</div>
        <div class="col-md-9 bg-light text-dark p-3">□ เนื้อหาหลัก</div>
    </div>
</div>

<!-- ตัวอย่าง 4: ใช้ .container-lg (Layout ขนาดใหญ่) -->
<div class="container-lg bg-light text-dark mt-4 p-4 border">
    <h2 class="text-center">□ ตัวอย่าง .container-lg</h2>
    <p class="text-center">เต็มจอจนถึง lg ขึ้นไป</p>
    <div class="row">
        <div class="col-lg-4 bg-danger text-white p-3">□ คอลัมน์ 1</div>
        <div class="col-lg-4 bg-primary text-white p-3">□ คอลัมน์ 2</div>
        <div class="col-lg-4 bg-success text-white p-3">□ คอลัมน์ 3</div>
    </div>
</div>

<!-- ตัวอย่าง 5: ใช้ .container-xl (Layout ใหญ่มาก) -->
<div class="container-xl bg-dark text-white mt-4 p-4">
    <h2 class="text-center">□ ตัวอย่าง .container-xl</h2>
    <p class="text-center">ใช้สำหรับหน้าจอที่ใหญ่ขึ้น</p>
    <div class="row">
        <div class="col-xl-6 bg-light text-dark p-3">□ คอลัมน์ 1</div>
        <div class="col-xl-6 bg-secondary text-white p-3">□ คอลัมน์ 2</div>
    </div>
</div>

<!-- Bootstrap JS -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>
```

```
</body>
</html>
```

อธิบายการทำงานของแต่ละส่วน

1. **ตัวอย่าง .container** ⟶ ขนาดจะปรับตาม breakpoint และอยู่ตรงกลาง
2. **ตัวอย่าง .container-fluid** ⟶ ใช้เต็มความกว้างของหน้าจอ (100%)
3. **ตัวอย่าง .container-md** ⟶ เป็น full width จนถึง md (768px) แล้วเปลี่ยนเป็น fixed
4. **ตัวอย่าง .container-lg** ⟶ เป็น full width จนถึง lg (992px) แล้วเปลี่ยนเป็น fixed
5. **ตัวอย่าง .container-xl** ⟶ เหมาะสำหรับหน้าจอใหญ่ (≥1200px)

ตัวอย่างโค้ด HTML เต็มรูปแบบเพิ่มเติม ที่แสดงการใช้ .container, .container-fluid และ .container-{breakpoint}

```
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ตัวอย่างเพิ่มเติมของ Bootstrap Container</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css">
</head>
<body>

    <!-- ตัวอย่าง 6: การใช้ .container สำหรับเนื้อหากลางหน้า -->
    <div class="container mt-4 p-4 border">
        <h2 class="text-center">ตัวอย่าง .container</h2>
        <p class="text-center">เนื้อหานี้จะมีขนาดคงที่และอยู่ตรงกลางหน้าจอ</p>
        <div class="row">
            <div class="col-md-4 bg-primary text-white p-3">คอลัมน์ 1</div>
            <div class="col-md-4 bg-success text-white p-3">คอลัมน์ 2</div>
            <div class="col-md-4 bg-danger text-white p-3">คอลัมน์ 3</div>
        </div>
    </div>
```

```
<!-- ตัวอย่าง 7: ใช้ .container-fluid สำหรับเฮดเดอร์แบบเต็มจอ -->
<div class="container-fluid bg-dark text-white text-center p-4 mt-4">
    <h2>□ Header แบบเต็มจอ</h2>
    <p>เนื้อหานี้จะเต็มจอเสมอ</p>
</div>

<!-- ตัวอย่าง 8: .container-lg ใช้ร่วมกับการ์ด -->
<div class="container-lg mt-4">
    <h2 class="text-center">□ ตัวอย่าง .container-lg กับ Card</h2>
    <div class="row">
        <div class="col-md-4">
            <div class="card">
                <div class="card-body">
                    <h5 class="card-title">□ การ์ด 1</h5>
                    <p class="card-text">เนื้อหาของการ์ด 1</p>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <div class="card-body">
                    <h5 class="card-title">□ การ์ด 2</h5>
                    <p class="card-text">เนื้อหาของการ์ด 2</p>
                </div>
            </div>
        </div>
        <div class="col-md-4">
            <div class="card">
                <div class="card-body">
                    <h5 class="card-title">□ การ์ด 3</h5>
                    <p class="card-text">เนื้อหาของการ์ด 3</p>
                </div>
            </div>
```

```
            </div>
        </div>
    </div>

    <!-- ตัวอย่าง 9: การใช้ .container-md ร่วมกับ Navbar -->
    <div class="container-md mt-4">
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <div class="container-fluid">
                <a class="navbar-brand" href="#">□ Navbar</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse" id="navbarNav">
                    <ul class="navbar-nav">
                        <li class="nav-item"><a class="nav-link" href="#">หน้าแรก</a></li>
                        <li class="nav-item"><a class="nav-link" href="#">เกี่ยวกับ</a></li>
                        <li class="nav-item"><a class="nav-link" href="#">ติดต่อ</a></li>
                    </ul>
                </div>
            </div>
        </nav>
    </div>

    <!-- ตัวอย่าง 10: .container-xl ใช้กับตารางข้อมูล -->
    <div class="container-xl mt-4">
        <h2 class="text-center">□ ตัวอย่าง .container-xl กับตาราง</h2>
        <table class="table table-bordered table-striped">
            <thead class="table-dark">
                <tr>
                    <th>#</th>
                    <th>ชื่อ</th>
                    <th>อีเมล</th>
                    <th>เบอร์โทร</th>
```

```
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>1</td>
                <td>สมชาย</td>
                <td>somchai@example.com</td>
                <td>081-234-5678</td>
            </tr>
            <tr>
                <td>2</td>
                <td>สมหญิง</td>
                <td>somying@example.com</td>
                <td>089-876-5432</td>
            </tr>
        </tbody>
    </table>
</div>

<!-- Bootstrap JS -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

☐ **คำอธิบายโค้ดแต่ละตัวอย่าง**

1. **ตัวอย่างที่ 6**
   - ใช้ .container เพื่อให้อยู่ตรงกลางของหน้า
   - ใช้ .row และ .col-md-4 แบ่งคอลัมน์เป็น 3 ส่วน
2. **ตัวอย่างที่ 7**
   - ใช้ .container-fluid สำหรับเฮดเดอร์เต็มจอ
   - ใช้ bg-dark text-white เพื่อเปลี่ยนสีพื้นหลังและข้อความ
3. **ตัวอย่างที่ 8**
   - ใช้ .container-lg ซึ่งจะเป็น **Fixed เมื่อหน้าจอใหญ่ขึ้น (≥992px)**

- ใช้ **Bootstrap Card Component** เพื่อจัดรูปแบบข้อมูล

4. **ตัวอย่างที่ 9**
   - ใช้ .container-md ร่วมกับ **Bootstrap Navbar**
   - เมนูจะขยายเมื่อหน้าจอใหญ่ขึ้น
5. **ตัวอย่างที่ 10**
   - ใช้ .container-xl รองรับตารางข้อมูล
   - ใช้ .table, .table-bordered, .table-striped, .table-dark เพื่อปรับแต่งตาราง

สรุป

ตัวอย่าง	ประเภท	ลักษณะการทำงาน
6	.container	ใช้เนื้อหาขนาดกลาง อยู่ตรงกลางจอ
7	.container-fluid	ใช้เต็มจอ 100% ตลอดเวลา
8	.container-lg	ใช้การ์ดข้อมูลและเป็น Fixed เมื่อ lg ขึ้นไป
9	.container-md	ใช้ร่วมกับ Navbar ให้ทำงาน Responsive
10	.container-xl	ใช้กับตารางข้อมูลรองรับหน้าจอใหญ่

Row & Column ใน Bootstrap

Row และ Column เป็นโครงสร้างหลักของ **Bootstrap Grid System** ซึ่งใช้จัดเรียงองค์ประกอบบนหน้าเว็บให้เป็นระเบียบและรองรับการปรับแต่งแบบ **Responsive**

1. Row (.row) คืออะไร?

- เป็น **Container** ที่ใช้สำหรับจัดกลุ่ม **Column** (.col-*)
- ต้องอยู่ภายใน .container หรือ .container-fluid
- ใช้ **Flexbox** ในการจัดวางองค์ประกอบให้อยู่ในแนวเดียวกัน

2. Column (.col-*) คืออะไร?

- ใช้แบ่งพื้นที่ภายใน .row เป็น **12 ส่วน**
- ขนาดของคอลัมน์สามารถกำหนดได้ เช่น .col-6 (6 ส่วนจาก 12 ส่วน)
- รองรับการปรับขนาดตาม **Breakpoint** เช่น .col-md-6, .col-lg-4

ตัวอย่างการใช้งาน Row & Column

1 ตัวอย่างพื้นฐานของ .row และ .col-*

```
<!DOCTYPE html>
```