

บทที่ 1: แนะนำรายวิชาและแนวคิดการพัฒนา เว็บสมัยใหม่

รายวิชา INF67-175: Web Development

หัวข้อหลัก:

- โครงสร้างเว็บไซต์สมัยใหม่
 - Frontend vs Backend
 - ตัวอย่าง Web Application
- 

1.1 บทนำ

ในยุคดิจิทัลปัจจุบัน เว็บไซต์ไม่ใช่เพียง “หน้าแสดงข้อมูล” อีกต่อไป แต่กลายเป็น “แอปพลิเคชันออนไลน์” ที่สามารถโต้ตอบกับผู้ใช้ ดึงข้อมูลจากฐานข้อมูลแบบเรียลไทม์ และเชื่อมต่อกับระบบต่าง ๆ ผ่าน **API**

การเรียนพัฒนาเว็บในรายวิชา **INF67-175: Web Development**

เน้นให้นักศึกษาเข้าใจทั้ง **Frontend** และ **Backend**

รวมถึงแนวคิดของ **เว็บยุคใหม่ (Modern Web App)** ที่ยืดหยุ่นและพร้อมใช้งานจริง

1.2 พัฒนาการของเว็บไซต์

ยุค	ลักษณะ	เทคโนโลยีหลัก	ตัวอย่าง
ยุคแรก (Static Web)	เว็บแบบคงที่ ไม่มีการโต้ตอบ	HTML, CSS	เว็บประชาสัมพันธ์
ยุคสอง (Dynamic Web)	เว็บมีระบบฐานข้อมูล ดึงข้อมูลอัตโนมัติ	PHP, MySQL, JavaScript	เว็บข่าว, เว็บบล็อก
ยุคสาม (Modern Web App)	เว็บไซต์ทำงานเหมือนแอปจริง มี API และอัปเดตทันที	HTML5, CSS3, JavaScript, Node.js, React	Facebook, Google Docs, LINE Web

1.3 แนวคิดของการพัฒนาเว็บสมัยใหม่

✓ 1. Responsive Design

เว็บไซต์ต้องสามารถปรับขนาดให้เหมาะกับหน้าจอทุกอุปกรณ์
ใช้เทคนิค CSS Framework เช่น **Bootstrap** หรือ **Tailwind CSS**




✓ 2. Interactive & Dynamic Content

เว็บไซต์ตอบสนองต่อผู้ใช้ได้ทันที เช่น ฟอรัมที่แสดงผลอัตโนมัติ
ใช้ JavaScript จัดการ DOM และ Event Handling



✓ 3. API Integration


เว็บไซต์ดึงข้อมูลจากบริการภายนอก เช่น

- ข้อมูลสภาพอากาศ (OpenWeather API)
 - ข่าวสาร
 - Google Sheet API
- 



✓ 4. Cloud Deployment

นำเว็บไซต์ขึ้นออนไลน์ได้ทันทีผ่านบริการฟรี เช่น
GitHub Pages, Netlify, Vercel




✓ 5. AI & Automation

ใช้เครื่องมือช่วยพัฒนา เช่น

ChatGPT, GitHub Copilot, Replit Ghostwriter

ช่วยสร้างโค้ดตัวอย่าง ลดเวลาในการพัฒนา



1.4 โครงสร้างของเว็บไซต์สมัยใหม่

เว็บไซต์สมัยใหม่มีองค์ประกอบหลัก 3 ส่วน ได้แก่

1. Frontend – ส่วนที่ผู้ใช้เห็น
2. Backend – ส่วนที่ประมวลผลหลังบ้าน
3. Database – ส่วนจัดเก็บข้อมูล



● 1. Frontend (ส่วนหน้า)

คือส่วนที่ผู้ใช้เห็นและโต้ตอบโดยตรง เช่น หน้าเว็บ ฟอรัม ปุ่ม รูปภาพ

ภาษา:

- HTML: โครงสร้าง
- CSS: การจัดรูปแบบ
- JavaScript: การโต้ตอบ

ตัวอย่าง:

```
<button onclick="alert('สวัสดี!')">คลิกฉัน</button>
```

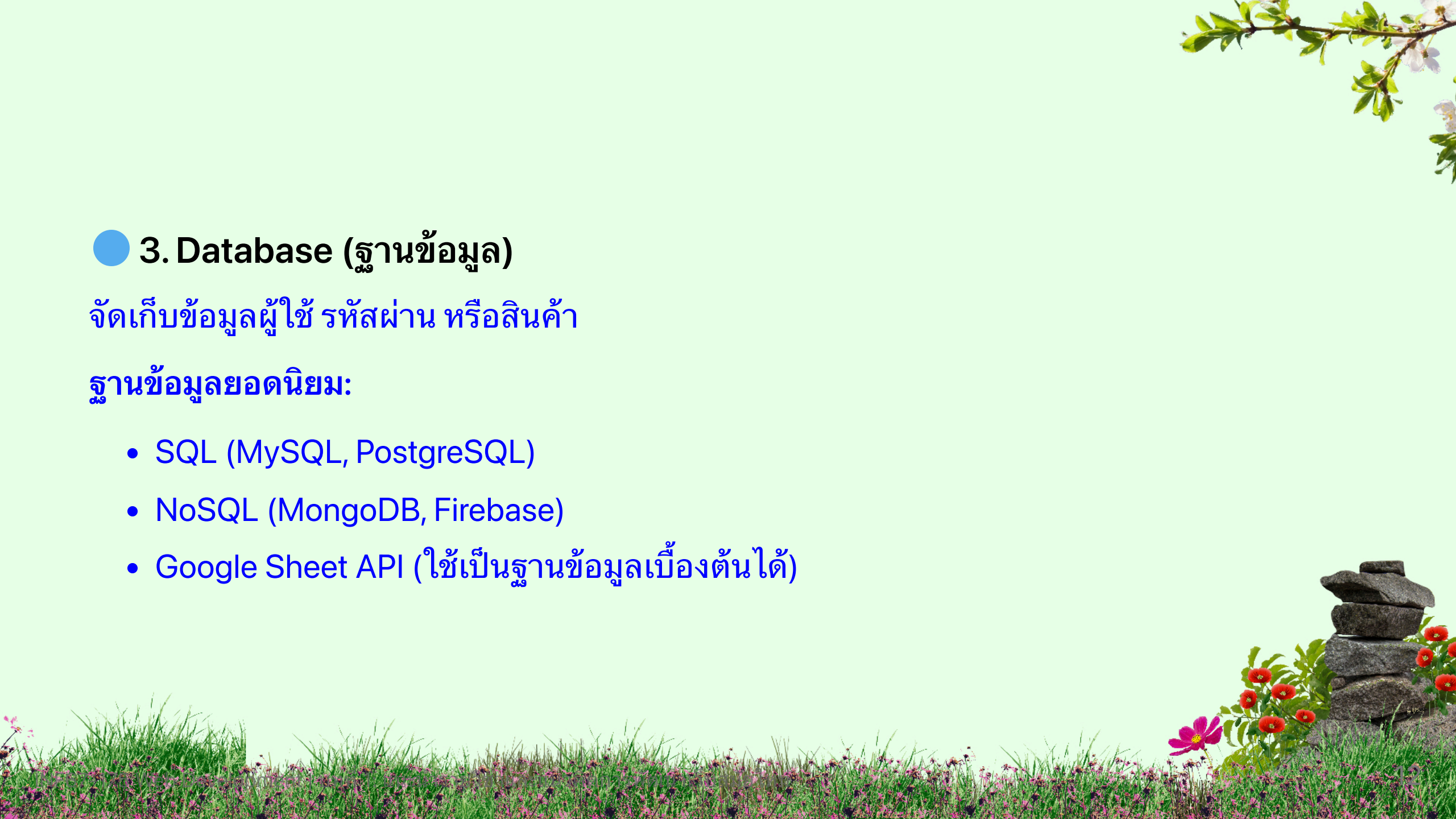
● 2. Backend (ส่วนหลัง)

อยู่บน Server ทำหน้าที่รับคำสั่ง ประมวลผล และส่งผลกลับมา

ภาษา: Node.js, Python, PHP

ตัวอย่าง (Node.js):

```
app.get("/hello", (req, res) => {  
  res.json({ message: "สวัสดีจาก Backend!" });  
});
```

● 3. Database (ฐานข้อมูล)

จัดเก็บข้อมูลผู้ใช้ รหัสผ่าน หรือสินค้า

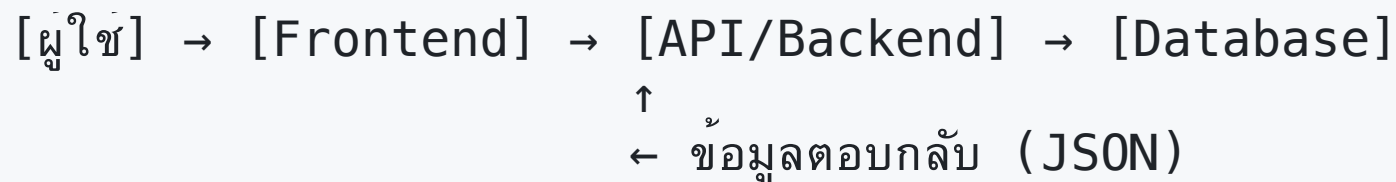
ฐานข้อมูลยอดนิยม:

- SQL (MySQL, PostgreSQL)
- NoSQL (MongoDB, Firebase)
- Google Sheet API (ใช้เป็นฐานข้อมูลเบื้องต้นได้)

1.5 การเชื่อมโยงระหว่าง Frontend และ Backend

การสื่อสารระหว่าง Frontend และ Backend ใช้ HTTP Request/Response ผ่าน RESTful API ที่ส่งข้อมูลแบบ JSON

แผนภาพกระบวนการ:



ตัวอย่างการเชื่อมต่อ API

```
fetch("https://jsonplaceholder.typicode.com/users/1")  
  .then(res => res.json())  
  .then(data => console.log(data));
```


1.6 ตัวอย่างเว็บแอปพลิเคชัน (Web Application)

🌐 ตัวอย่าง: ระบบ "Profile Info"

ผู้ใช้กรอก "User ID" → ระบบดึงข้อมูลจาก API → แสดงข้อมูลผู้ใช้

```
<form id="profileForm">
  <label>ใส่ User ID: </label>
  <input type="number" id="userId" min="1" max="10">
  <button type="submit">ดึงข้อมูล</button>
</form>

<div id="result"></div>
```

1.6 ตัวอย่างเว็บแอปพลิเคชัน (Web Application)

🌐 ตัวอย่าง: ระบบ "Profile Info"

```
<script>
document.getElementById("profileForm").addEventListener("submit", async (e) => {
  e.preventDefault();
  const id = document.getElementById("userId").value;
  const res = await fetch(`https://jsonplaceholder.typicode.com/users/${id}`);
  const data = await res.json();
  document.getElementById("result").innerHTML = `
    <h3>${data.name}</h3>
    <p>${data.email}</p>
    <p>${data.address.city}</p>
  `;
});
</script>
```

1.6 ตัวอย่างเว็บแอปพลิเคชัน (Web Application)

🌐 ตัวอย่าง: ระบบ "Profile Info"

📖 **Frontend:** แสดงฟอร์มและผลลัพธ์

📖 **Backend:** API ที่ให้ข้อมูล


📖 **ผลลัพธ์:** หน้าเว็บ Interactive แบบทันที

1.7 เครื่องมือที่นิยมใช้ในการพัฒนาเว็บ

ประเภท	เครื่องมือแนะนำ
Code Editor	Visual Studio Code
Version Control	Git / GitHub
Frontend Framework	Bootstrap, React
Backend Framework	Node.js, Express
API Tools	Postman, Thunder Client
Deploy Website	GitHub Pages, Netlify
DevTools	Chrome Developer Tools

1.8 สรุปบทเรียน

หัวข้อ	สิ่งที่ควรเข้าใจ
Modern Web	เว็บไซต์ต้อง Responsive และ Dynamic
โครงสร้างเว็บ	แบ่งเป็น Frontend / Backend / Database
การเชื่อมต่อ	ใช้ API ส่งข้อมูลระหว่างผู้ใช้กับเซิร์ฟเวอร์
ตัวอย่างจริง	เว็บไซต์ส่วนใหญ่เป็น Web App
การพัฒนา	ใช้เครื่องมือ เช่น VS Code, GitHub, AI Tools



กิจกรรมถ่ายทอด


ใบงานที่ 1:

สำรวจเว็บไซต์จริง 1 เว็บไซต์ (เช่น Shopee, Line Web, Google Docs)
อธิบายว่าเว็บไซต์นั้นมีส่วนของ **Frontend** และ **Backend** อย่างไร
และมีการใช้ API หรือไม่



คำถามทบทวน

1. เว็บไซต์สมัยใหม่แตกต่างจากเว็บยุคเก่าอย่างไร?
2. Frontend และ Backend ทำหน้าที่ต่างกันอย่างไร?
3. API คืออะไร และมีประโยชน์อย่างไรในเว็บแอป?
4. ยกตัวอย่างเครื่องมือที่ใช้ในแต่ละส่วนของการพัฒนาเว็บ
5. ทำไมการ Deploy เว็บไซต์บน Cloud จึงสำคัญ?



สรุปท้ายบท

การพัฒนาเว็บยุคใหม่คือการผสานทั้ง **Frontend** และ **Backend**
ให้สามารถโต้ตอบ ดึงข้อมูล และแสดงผลแบบเรียลไทม์
พร้อมต่อยอดเป็น **Web Application** ที่พร้อมใช้งานจริงในโลกออนไลน์

