**การพัฒนาแพลตฟอร์สำหรับการเผยแพร่ผลงานและจ้างงานด้านดนตรี**

**A Development of a Platform for Publishing and Hiring Musical Works**

**นาย อภิสิทธิ์ สุขคำชา**

**แบบเสนอหัวข้อโครงงานนักศึกษา**

**แบบเสนอนี้เป็นส่วนหนึ่งของวิชาสัมมนาเทคโนโลยีสารสนเทศ (4134902)**

**ตามหลักสูตร**

**วิทยาศาสตรบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ**

**มหาวิทยาลัยราชภัฏบุรีรัมย์ ปี**

**การศึกษา 2567**

**แบบเสนอโครงงานนักศึกษา**

**(Senior Project Proposal)**

**สาขาวิชาเทคโนโลยีสารสนเทศ กลุ่มวิชา เครือข่าย**

**1.ชื่อโครงงาน**

ชื่อภาษาไทย : การพัฒนาแพลตฟอร์สำหรับการเผยแพร่ผลงานและจ้างงานด้านดนตรี

ชื่อภาษาอังกฤษ : A Development of a Platform for Publishing and Hiring Musical Works

**2.ขื่อผู้เสนอโครงงาน**

นายอภิสิทธิ์ สุขคำชา รหัสประจำตัว 650112418071

Mr.Apisit Sukkamcha Student ID 650112418071

**3.อาจารย์ที่ปรึกษาโครงงาน**

อาจารย์ ผู้ช่วยศาสตราจารย์ดรัสวิน วงศ์ปรเมษฐ์

ตำแหน่ง อาจารย์ประจำสาขาวิชาเทคโนโลยีสารสนเทศ

หน่วยงานสาขาวิชาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์

มหาวิทยาลัยราชภัฏบุรีรัมย์

**4.ที่มาและความสำคัญของปัญหา**

ในปัจจุบัน วงการดนตรีมีการเติบโตอย่างรวดเร็ว โดยเฉพาะในกลุ่มนักดนตรีอิสระที่มีความสามารถในการสร้างสรรค์ผลงานที่หลากหลาย แต่กลับประสบปัญหาขาดพื้นที่ที่เหมาะสมในการเผยแพร่ผลงาน แลกเปลี่ยนความรู้กับเพื่อนร่วมวงการ หรือสร้างเครือข่ายกับคู่ค้าที่สนใจซื้อขายเพลงและทำงานร่วมกัน นอกจากนี้ นักดนตรีจำนวนไม่น้อยยังเผชิญกับความท้าทายในการสร้างเอกลักษณ์เฉพาะตัวให้กับผลงานของตน ซึ่งส่งผลกระทบต่อความสามารถในการแข่งขันในตลาดดนตรีที่มีการแข่งขันสูง (Boehm, 2007; Hracs et al., 2013) การพัฒนาแพลตฟอร์มออนไลน์ที่ทำหน้าที่เป็นคอมมูนิตี้สำหรับนักดนตรี โดยนำเสนอฟังก์ชันอัพโหลดและแสดงผลงานดนตรีเพื่อแบ่งปันผลงานให้กับกลุ่มนักดนตรีและผู้ฟังที่สนใจ ระบบแชทและการสร้างเครือข่ายซึ่งอำนวยความสะดวกในการเจรจาซื้อขายเพลงหรือแลกเปลี่ยนความรู้ระหว่างนักดนตรี การใช้งาน AI เพื่อสร้างเสียงดนตรีอัตโนมัติช่วยสร้างเสียงดนตรีต้นแบบ หรือเพิ่มความแปลกใหม่ให้กับผลงาน เพื่อช่วยนักดนตรีพัฒนาผลงานที่มีเอกลักษณ์ (Herremans et al., 2017)

จากสภาพปัญหาข้างต้นจึงมีความจำเป็นในการใช้เทคโนโลยีสารสนเทศนในการจัดการประกอบด้วย เว็บไซต์ซึ่งพัฒนาด้วย Next.js สำหรับส่วนหน้าเว็บที่เน้นประสิทธิภาพสูงและการตอบสนองที่รวดเร็ว MongoDB สำหรับการจัดการฐานข้อมูลที่สามารถรองรับการทำงานร่วมกันในหลายมิติ และ โมเดล Machine Learning สำหรับสร้างและปรับแต่งเสียงดนตรี (Goodfellow et al., 2014) นักดนตรีอิสระในยุคปัจจุบันต้องการแพลตฟอร์มที่เป็นพื้นที่กลางสำหรับเผยแพร่ผลงานและแลกเปลี่ยนความคิดเห็น (Hracs et al. ,2013)

เพื่อตอบสนองความต้องการดังกล่าว โครงการพัฒนาแพลตฟอร์มเผยแพร่ผลงานและจ้างงานด้านดนตรีจะช่วยส่งเสริมการสร้างเครือข่ายระหว่างนักดนตรีและผู้สนใจในอุตสาหกรรม อีกทั้งยังช่วยลดต้นทุนด้านการผลิตเสียงดนตรีด้วยการใช้ AI ช่วยเร่งกระบวนการสร้างสรรค์ นอกจากนี้ยังเปิดโอกาสให้ศิลปินอิสระสามารถขยายธุรกิจและพัฒนาความสัมพันธ์กับผู้ร่วมงานและผู้ฟัง การพัฒนาแพลตฟอร์มนี้คาดว่าจะช่วยสร้างพื้นที่ที่ส่งเสริมการเติบโตของนักดนตรีอิสระในไทยและในระดับนานาชาติ โดยการเชื่อมโยงผู้มีส่วนเกี่ยวข้องในวงการดนตรี ทั้งนักดนตรี ผู้ผลิตเพลง และผู้ฟัง ให้สามารถแลกเปลี่ยนความคิดและทำงานร่วมกันได้ง่ายขึ้น โครงการนี้ยังคาดว่าจะช่วยยกระดับวงการดนตรีไทยไปในทิศทางที่สร้างสรรค์และยั่งยืน

**5.วัตถุประสงค์ของโครงงาน**

5.1 เพื่อสร้างแนวทางในการใช้งานโมเดลด้านปัญญาประดิษฐ์สำหรับสร้างดนตรี

5.2 เพื่อพัฒนาแพลตฟอร์มเผยแพร่ผลงานและจ้างงานด้านดนตรี

**6.ประโยชน์ที่คาดว่าจะได้รับ**

6.1 ได้สร้างแนวทางในการใช้งานโมเดลด้านปัญญาประดิษฐ์สำหรับสร้างดนตรี

6.2 ได้พัฒนาแพลตฟอร์มเผยแพร่ผลงานและจ้างงานด้านดนตรี

6.3 โปรดิวเซอร์จะมีพื้นที่สำหรับแสดงผลงานและสร้างเครือข่ายกับเพื่อนร่วมวงการได้สะดวกยิ่งขึ้น

6.4 ผู้ใช้ทั่วไปสามารถค้นหาผลงานเพลงและติดต่อซื้อขายเพลงกับโปรดิวเซอร์ได้ง่ายและปลอดภัย

6.5 ได้ระบบแชทช่วยเพิ่มโอกาสในการเจรจาซื้อขายผลงานและการแลกเปลี่ยนความรู้ระหว่างนักดนตรี

6.6 ได้ระบบปัญญาประดิษฐ์สร้างเสียงดนตรีจะช่วยลดเวลาและทรัพยากรในการสร้างสรรค์ผลงานเพลงใหม่

6.7 สร้างคอมมูนิตี้ที่เข้มแข็งและยั่งยืนสำหรับคนในวงการดนตรีที่สามารถแลกเปลี่ยนความรู้และไอเดียได้ตลอดเวลา

**7.ขอบเขตของโครงงาน**

พัฒนาแพลตฟอร์มเผยแพร่ผลงานและจ้างงานด้านดนตรีมีขอบเขตของโครงงาน ดังนี้

7.1 ขอบเขตของโปรดิวเซอร์

7.1.1 สมัครสมาชิก

7.1.1.1 ชื่อผู้ใช้

7.1.1.2 อีเมล

7.1.1.3 รหัสผ่าน

7.1.1.4 คำอธิบาย (ไม่จำเป็นต้องระบุ)

7.1.1.5 อัพโหลดรูปโปรไฟล์

7.1.1.6 เบอร์โทร

7.1.1.7 role (Producer)

7.1.1.8 URLYoutube (ไม่จำเป็นต้องระบุ)

7.1.1.9 URLInstargram (ไม่จำเป็นต้องระบุ)

7.1.1.10 URLFacebook (ไม่จำเป็นต้องระบุ)

7.1.2 เข้าสู่ระบบ

7.1.2.1 ชื่อผู้ใช้

7.1.2.2 รหัสผ่าน

7.1.3 โพสต์ข้อมูลดนตรีโปรดิวเซอร์

7.1.3.1 แนวดนตรี (Hiphop,pop,rock,อื่นๆ)

7.1.3.2 ความเร็วของดนตรี (BPM)

7.1.3.3 คีย์ของดนตรี (B,D,G,อื่นๆ)

7.1.3.4 ไฟล์ดนตรี (instrumental)

7.1.3.5 รูปปกดนตรี

7.1.4 โปรไฟล์โปรดิวเซอร์

7.1.4.1 แสดงรูปโปรไฟล์

7.1.4.2 แสดงชื่อผู้ใช้

7.1.4.3 แสดงเบอร์โทร

7.1.4.4 แสดงอีเมล

7.1.4.5 แสดง role (Producer)

7.1.4.6 ปุ่ม URLYoutube

7.1.4.7 ปุ่ม URLInstargram

7.1.4.8 ปุ่ม URLFacebook

7.1.4.9 แสดง คำอธิบาย

7.1.4.10 ปุ่ม สนทนา (ระบบแชท)

7.1.4.11 แสดงผลงานดนตรี

7.1.5 ระบบแชท

7.1.5.1 สามารถพูดคุยกับผู้ใช้อื่นๆได้ทั้งหมด

7.1.5.2 สามารถส่งไฟล์รูปภาพได้

7.1.5.3 สามารถส่งลิ้งต่างๆได้ เช่น googleDrive,youtube,อื่นๆ

7.1.6 ค้นหาข้อมูลเพลงและดนตรีของผู้ใช้ทั่วไปและโปรดิวเซอร์

7.1.6.1 ชื่อเพลง

7.1.6.2 แนวเพลง

7.1.6.3 คีย์ของเพลง

7.1.6.4 ความเร็วของเพลง

7.2 ขอบเขตของผู้ใช้ทั่วไป

7.2.1 สมัครสมาชิก

7.2.1.1 ชื่อผู้ใช้

7.2.1.2 อีเมล

7.2.1.3 รหัสผ่าน

7.2.1.4 คำอธิบาย (ไม่จำเป็นต้องระบุ)

7.2.1.5 อัพโหลดรูปโปรไฟล์

7.2.1.6 เบอร์โทร

7.2.1.7 role (RegularUser)

7.2.1.8 URLYoutube (ไม่จำเป็นต้องระบุ)

7.2.1.9 URLInstargram (ไม่จำเป็นต้องระบุ)

7.2.1.10 URLFacebook (ไม่จำเป็นต้องระบุ)

7.2.2 เข้าสู่ระบบ

7.2.2.1 ชื่อผู้ใช้

7.2.2.2 รหัสผ่าน

7.2.3 ค้นหาข้อมูลเพลงและดนตรีของผู้ใช้ทั่วไปและโปรดิวเซอร์

7.2.3.1 ชื่อเพลง

7.2.3.2 แนวเพลง

7.2.3.3 คีย์ของเพลง

7.2.3.4 ความเร็วของเพลง

7.2.4 สร้างเสียงดนตรีด้วยปัญญาประดิษฐ์

7.2.4.1 แนวเพลง

7.2.4.2 ความเร็วของเพลง

7.2.4.3 คีย์ของเพลง

7.2.4.4 เนื้อเพลง (ไม่จำเป็นต้องระบุ)

7.2.5 โปรไฟล์ผู้ใช้ทั่วไป

7.2.5.1 แสดงรูปโปรไฟล์

7.2.5.2 แสดงชื่อผู้ใช้

7.2.5.3 แสดงเบอร์โทร

7.2.5.4 แสดงอีเมล

7.2.5.5 แสดง role (RegularUser)

7.2.5.6 ปุ่ม URLYoutube

7.2.5.7 ปุ่ม URLInstargram

7.2.5.8 ปุ่ม URLFacebook

7.2.5.9 แสดง คำอธิบาย

7.2.5.10 ปุ่มพูดคุย(ระบบแชท)

7.2.5.11 แสดงผลงานเพลง

7.2.6 ระบบแชท

7.2.6.1 สามารถพูดคุยกับผู้ใช้อื่นๆได้ทั้งหมด

7.2.6.2 สามารถส่งไฟล์รูปภาพได้

7.2.6.3 สามารถส่งลิ้งต่างๆได้ เช่น googleDrive,youtube,อื่นๆ

7.2.7 โพสต์ข้อมูลเพลงของผู้ใช้ทั่วไป

7.2.7.1 ชื่อเพลง

7.2.7.2 ปกเพลง

7.2.7.3 เนื้อเพลง (ถ้ามี)

7.2.7.4 ไฟล์เพลง (mp3,wav)

7.3 ขอบเขตของแอดมิน

7.3.1 เข้าสู่ระบบ

7.3.1.1 ชื่อผู้ใช้

7.3.1.2 รหัสผ่าน

7.3.2 จัดการลบโพสต์ที่หรือบัญชีผู้ใช้ที่ผิดกฎของ

7.3.2.1 ชื่อผู้ใช้

7.3.2.2 ลบโพสต์ (\*\*สำหรับผู้ใช้ที่ทำผิดกฎ\*\*)

7.3.2.3 ลบบัญชีผู้ใช้ (\*\*สำหรับผู้ใช้ที่ทำผิดกฎ\*\*)

7.3.3 โปรไฟล์แอดมิน

7.3.3.1 แสดงรูปโปรไฟล์

7.3.3.2 แสดงชื่อผู้ใช้

7.3.3.5 แสดง role (admin)

7.3.3.10 ปุ่มสนทนา(ระบบแชท)

7.3.4 ระบบแชท

7.3.4.1 สามารถพูดคุยกับผู้ใช้อื่นๆได้ทั้งหมด

7.3.4.2 สามารถส่งไฟล์รูปภาพได้

7.3.4.3 สามารถส่งลิ้งต่างๆได้ เช่น googleDrive,youtube,อื่นๆ

**8.ทฤษฎีและงานวิจัยที่เกี่ยวข้อง**

8.1 ปัญญาประดิษฐ์สำหรับการสร้างดนตรี

กำลังมีบทบาทสำคัญในการเปลี่ยนแปลงวงการดนตรี โดยเฉพาะในด้านการสร้างสรรค์เพลงและการจัดการข้อมูลที่เกี่ยวข้องกับดนตรี เช่น โมเดล MuseNet ของ OpenAI ที่สามารถสร้างเพลงต้นฉบับโดยวิเคราะห์รูปแบบและสไตล์จากเพลงที่มีอยู่ (Shaip, 2024) นอกจากนี้ การใช้ AI ยังช่วยให้ผู้ใช้สามารถสร้างเพลงได้อย่างรวดเร็วและง่ายดาย โดยไม่จำเป็นต้องมีความรู้ด้านดนตรีมากนัก (MRArranger, 2023). โดยแนวคิดหลักในการประพันธ์เพลงทั่วไปมีหลักการทำงานของวิทยาการคอมพิวเตอร์พื้นฐานที่เป็นรากฐานในการพัฒนาปัญญาประดิษฐ์ โดยแบ่งออกเป็น 3 กระบวน (movement) ได้แก่

1. “Algorithm” คือ ขั้นตอนวิธีการแก้ปัญหาทางคอมพิวเตอร์

2. “Heuristic” คือ การแก้ปัญหาแบบศึกษาสำนึกของมนุษย์

3. “Debug” คือ การแก้ไขจุดบกพร่องทางโปรแกรม

โครงสร้างหลักของเพลง 3 กระบวนคือ เร็ว-ช้า-เร็ว ตามแบบแผนที่เหมาะสมกับเรื่องราวที่ผู้ประพันธ์ต้องการจะนำเสนอ โดยแต่ละท่อนเสนอลักษณะพื้นผิวของดนตรี (texture Music) ที่แตกต่างออกไป โดยคำนึงถึงองค์ประกอบสำคัญที่ใช้ในการประพันธ์เพลงได้แก่ จังหวะ (rhythm) จังหวะในบทเพลงจะพบการใช้จังหวะขัดแบบโพลีริธึม (polyrhythm) ตลอดทั่วทั้งเพลง เพื่อให้เกิดความซับซ้อนในรูปแบบที่ผู้ประพันธ์เห็นว่ามีความสอดคล้องกับบรรยากาศของการทำงานของปัญญาประดิษฐ์ตามจินตนาการของผู้ประพันธ์ และช่วยทำให้การบรรเลงเวียนซ้ำอันเป็นเอกลักษณ์ให้เพลงมีความน่าสนใจมากขึ้น ความถี่ของเสียง (frequency) ในบทเพลงผู้ประพันธ์เลือกใช้ระดับเสียงไดอาโทนิค 7 โน้ตธรรมดาที่อยู่ในโมด (mode) ต่างๆ ผู้ประพันธ์ไม่ต้องการให้เกิดความซับซ้อนของความถี่ของเสียงมาก เพราะได้เน้นให้เกิดความซับซ้อนไปที่องค์ประกอบจังหวะมากแล้ว ความหนาแน่นของเสียง (density) ผู้ประพันธ์ต้องการสร้างความหนาแน่นของเสียงที่เกิดจากการขัดกันของจังหวะแบบโพลิริธึมที่ดำเนินอยู่ในหลายแนวเสียงพร้อมๆ กัน จึงเน้นไปที่การใช้เทคนิคการนำออสตินาโตจังหวะมาบรรเลงซ้อนทับกัน รวมถึงการสอดประสานของทำนองเพื่อให้เกิดความซับซ้อน สร้างความหนาแน่นจากการบรรเลงซ้อนทับกันของช่วงทำนองเวียนซ้ำ สีสันของเครื่องดนตรี (timbre) ในบทเพลงมีการใช้เครื่องดนตรีที่มีแหล่งกำเนิดเสียงแบบสังเคราะห์จากคลื่นไฟฟ้าซินธิไซเซอร์ ผู้ประพันธ์ต้องการนำเสนอเอกลักษณ์ของบทเพลงที่เกี่ยวกับเทคโนโลยี บรรเลงร่วมกับเครื่องดนตรีอื่นโดยมีการออกและการประสานเสียง (harmony) ในบทเพลงจะใช้การประสานเสียงตามแบบดนตรีโทนาล (Tonal) ผสมผสานกับการประสานเสียงแบบโมด (Mode) โดยจะไม่เน้นไปที่การเคลื่อนคอร์ดมากนัก แต่จะเน้นการเวียนซ้ำอยู่กับที่และค่อยๆ พัฒนาช่วงทำนองเวียนซ้ำอย่างต่อเนื่อง (ธาวิน ไล้ทอง, 2562)



**รูปที่ 1** ตัวอย่างโน้ตที่ใช้สำหรับงานดนตรี

8.2 แนวทางการพัฒนาแพลตฟอร์ม

การพัฒนาแพลตฟอร์มที่เหมาะสมในการเผยแพร่ผลงานและจ้างงานด้านดนตรี สามารถนำเสนอฟังก์ชันที่ช่วยให้ผู้ใช้งานสามารถอัปโหลดผลงานดนตรีของตนเอง พร้อมระบบ AI ที่ช่วยในการสร้างเสียงและเรียบเรียงเพลง เช่น Boomy AI และ Suno ที่มีฟังก์ชันการสร้างเพลงอัตโนมัติและการช่วยในการแต่งเพลง (MRArranger, 2023) นอกจากนี้ การใช้โมเดล AI เช่น Fugatto ของ Nvidia ที่สามารถสร้างเสียงและปรับแต่งเสียงได้ตามคำสั่ง จะช่วยเพิ่มความหลากหลายในการผลิตเสียงดนตรี (Nvidia, 2024).

8.2.1 วงจรการพัฒนาระบบ SDLC (Software Development Life Cycle)

เป็นแนวทางที่สำคัญในอุตสาหกรรมซอฟต์แวร์ ซึ่งช่วยให้การออกแบบ พัฒนา และจัดการซอฟต์แวร์เป็นไปอย่างมีระบบและมีประสิทธิภาพ SDLC เป็นกระบวนการที่มีขั้นตอนชัดเจน ซึ่งประกอบด้วยการวางแผน การวิเคราะห์ความต้องการ การออกแบบ การพัฒนา การทดสอบ การติดตั้ง และการบำรุงรักษาซอฟต์แวร์ โดยทุกขั้นตอนมีเป้าหมายในการสร้างซอฟต์แวร์ที่มีคุณภาพสูงและตอบสนองความต้องการของผู้ใช้งาน (Skooldio, 2023) ในขั้นตอนแรกของ SDLC คือ การวิเคราะห์ความต้องการ (Requirement Analysis) ซึ่งทีมพัฒนาจะทำการรวบรวมข้อมูลจากผู้มีส่วนได้ส่วนเสีย เช่น ลูกค้า ผู้ใช้และผู้เชี่ยวชาญ เพื่อกำหนดความต้องการที่ชัดเจนสำหรับซอฟต์แวร์ที่จะพัฒนา ขั้นตอนนี้เป็นสิ่งสำคัญเนื่องจากความเข้าใจที่ถูกต้องเกี่ยวกับความต้องการจะช่วยลดความเสี่ยงในการพัฒนาซอฟต์แวร์ในภายหลัง (Tech Terrotor, 2021) หลังจากนั้นจะเป็นขั้นตอน การออกแบบซอฟต์แวร์ (Software Design) ซึ่งทีมพัฒนาจะสร้างสถาปัตยกรรมของระบบ โดยอาจเลือกใช้รูปแบบต่าง ๆ เช่น Monolithic หรือ Microservices ขึ้นอยู่กับลักษณะของโปรเจ็กต์และข้อกำหนดทางธุรกิจ ในขั้นตอนนี้จะมีการกำหนดเทคโนโลยี เครื่องมือ และวิธีการที่จะใช้ในการพัฒนาซอฟต์แวร์ (AWS, 2024) เมื่อเสร็จสิ้นขั้นตอนการออกแบบ ทีมพัฒนาจะเข้าสู่ขั้นตอน การพัฒนาซอฟต์แวร์ (Software Development) โดยเริ่มเขียนโค้ดตามที่ได้ออกแบบไว้ ทีมงานจะใช้ภาษาโปรแกรมและเครื่องมือที่เหมาะสมในการสร้างฟังก์ชันต่าง ๆ ของซอฟต์แวร์ ในระหว่างนี้จะมีการตรวจสอบและทดสอบฟังก์ชันย่อยเพื่อให้แน่ใจว่าทุกอย่างทำงานได้ตามที่คาดหวัง (TechTerrotor, 2021) ต่อมาเป็นขั้นตอน การทดสอบซอฟต์แวร์ (Software Testing) ซึ่งเป็นกระบวนการที่สำคัญในการตรวจสอบข้อผิดพลาดและปัญหาที่อาจเกิดขึ้นในซอฟต์แวร์ ทีมทดสอบจะทำงานร่วมกับทีมพัฒนาเพื่อตรวจสอบว่าโปรแกรมทำงานตามความต้องการและไม่มีข้อผิดพลาด ก่อนที่จะนำไปติดตั้งใช้งานจริง (AWS, 2023) เมื่อผ่านขั้นตอนทดสอบแล้ว จะเข้าสู่ขั้นตอน การติดตั้งและนำซอฟต์แวร์ไปใช้งาน (Software Deployment) ในขั้นตอนนี้ ซอฟต์แวร์ที่ได้รับการพัฒนาและทดสอบเสร็จสมบูรณ์จะถูกนำไปติดตั้งในสภาพแวดล้อมจริง เพื่อให้ผู้ใช้สามารถเข้าถึงและใช้งานได้อย่างเต็มประสิทธิภาพ (Skooldio, 2024) สุดท้ายคือขั้นตอน การบำรุงรักษาซอฟต์แวร์ (Software Maintenance) ซึ่งเป็นกระบวนการดูแลรักษาซอฟต์แวร์หลังจากที่ได้ถูกติดตั้งใช้งานแล้ว ในระหว่างนี้อาจเกิดปัญหาหรือข้อจำกัดใหม่ ๆ ที่ต้องมีการแก้ไขหรือปรับปรุงเพื่อให้ซอฟต์แวร์ทำงานได้อย่างต่อเนื่องและตอบสนองต่อความต้องการของผู้ใช้ (TechTerrotor, 2021) SDLC ยังมีโมเดลต่าง ๆ ที่สามารถนำมาใช้ในการพัฒนาซอฟต์แวร์ เช่น Waterfall Model, Agile Model, Spiral Model และ V-Model โดยแต่ละโมเดลมีลักษณะเฉพาะและเหมาะสมกับประเภทของโปรเจ็กต์ที่แตกต่างกัน การเลือกโมเดลที่เหมาะสมจะช่วยให้กระบวนการพัฒนาซอฟต์แวร์เป็นไปอย่างราบรื่นและมีประสิทธิภาพมากขึ้น (Panisa, 2024)

โดยจะมีขั้นตอนการพัฒนาระบบดังนี้

1) การกำหนดปัญหา (Requirement Analysis/Problem Definition)

ขั้นตอนนี้เป็นการรวบรวมและทำความเข้าใจปัญหาหรือความต้องการของระบบใหม่ที่ต้องการพัฒนา โดยรวมถึงการพูดคุยกับผู้มีส่วนเกี่ยวข้อง (stakeholders) เพื่อระบุเป้าหมาย ขอบเขตของระบบ และข้อจำกัดต่างๆ

2) วิเคราะห์ระบบ (System Analysis)

ในขั้นนี้จะวิเคราะห์ความต้องการเชิงลึกที่ได้จากขั้นแรก เพื่อตรวจสอบความเป็นไปได้เชิงเทคนิคและเศรษฐศาสตร์ และออกแบบโครงสร้างข้อมูลและกระบวนการทำงานของระบบ รวมถึงการระบุความเสี่ยงที่อาจเกิดขึ้น

3) การออกแบบ (System Design)

การออกแบบระบบเป็นการวางโครงสร้างรายละเอียดของระบบ เช่น การออกแบบอินเทอร์เฟซ (UI/UX) การออกแบบโครงสร้างฐานข้อมูล และการกำหนดรูปแบบการเชื่อมโยงของส่วนประกอบต่างๆ เพื่อให้ทีมพัฒนาเข้าใจชัดเจนถึงสิ่งที่จะต้องพัฒนา

4) การพัฒนาและการทดสอบ (Development and Testing)

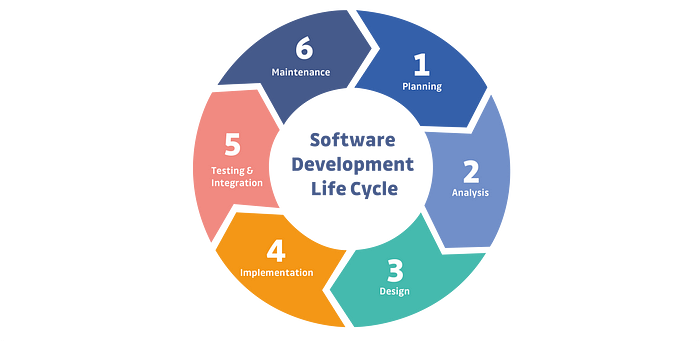
ทีมพัฒนาจะเริ่มเขียนโค้ดตามการออกแบบที่ได้เตรียมไว้ โดยใช้เครื่องมือและเทคโนโลยีที่เหมาะสม เมื่อการพัฒนาสิ้นสุด จะดำเนินการทดสอบเพื่อหาข้อผิดพลาด (bugs) หรือปัญหาต่างๆ เพื่อปรับปรุงระบบก่อนเปิดใช้งาน

5) ติดตั้งระบบ (Implementation/Deployment)

ระบบที่พัฒนาเสร็จและผ่านการทดสอบจะถูกติดตั้งและใช้งานจริงในสภาพแวดล้อมการทำงานขององค์กร รวมถึงการฝึกอบรมผู้ใช้งานและการปรับปรุงระบบเพื่อให้เหมาะสมกับการใช้งาน

6) การประเมินประสิทธิภาพการทำงานของระบบ (Maintenance/Evaluation)

หลังจากระบบเริ่มใช้งานจริง จะมีการตรวจสอบประสิทธิภาพอย่างต่อเนื่อง รวมถึงการอัปเดต ปรับปรุง หรือแก้ไขปัญหาเพื่อให้ระบบสามารถรองรับความต้องการของผู้ใช้และสภาพแวดล้อมที่เปลี่ยนแปลงได้ในอนาคต



**ภาพที่ 1** Software Development System : SDLC

**ที่มา :** ธนกฤต คล้ายสังข์ (2567 : 2)

8.2.2 แผนภาพกระแสข้อมูล

โอภาส (2548: 163-165) ได้กล่าวถึงความหมายและความเป็นมาของแผนภาพกระแสข้อมูล ไว้ว่า แผนภาพกระแสข้อมูล (Data Flow Diagram : DFD) เป็นแบบจําลองที่นํามาใช้กับการวิเคราะห์และออกแบบระบบเชิงโครงสร้าง ที่มีการนํามาใช้ตั้งแต่ยุคที่มีการเริ่มใช้ภาษาระดับสูงอย่างภาษาโคบอลโดยแผนภาพกระแสข้อมูลจะแสดงความสัมพันธ์ระหว่างโปรเซส (Processing) กับข้อมูล (Data) ที่เกี่ยวข้องโดยข้อมูลในแผนภาพจะทําให้ทราบว่าข้อมูลมาจากไหน, ข้อมูลไปที่ไหน, ข้อมูลเก็บไว้ที่ใด หรือเกิดเหตุการณ์ใดกับข้อมูลระหว่างทาง มีขั้นตอนในการวิเคราะห์เพื่อสร้างแผนภาพกระแสข้อมูล ดังนี้

ขั้นตอนที่ 1 ศึกษารูปแบบการทํางานทางฟิสิคัล (Physical) ของระบบงานเดิม

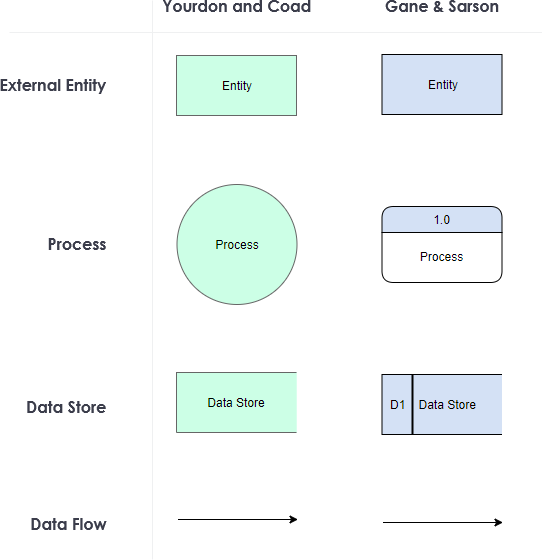
ขั้นตอนที่ 2 วิเคราะห์เพื่อให้ได้มาซึ่งแบบจําลองทางลอจิคัล (Logical) ของระบบงานเดิม

ขั้นตอนที่ 3 นําแบบจําลองทางลอจิคัลที่ได้จากขั้นตอนที่ 2 มาเพิ่มเติมความต้องการใหม่เข้าไปด้วยการปรับปรุงเพื่อเป็นแบบจําลองลอจิคัลของระบบงานใหม่

ขั้นตอนที่ 4 พัฒนาระบบงานใหม่ในรูปแบบของแบบจําลองฟิสิคัล

สัญลักษณ์ที่ใช้ในแผนภาพกระแสข้อมูล

สัญลักษณ์ในการเขียนแผนภาพกระแสข้อมูล ที่นิยมใช้กันในปัจจุบันมาจาก 2 องค์กร ได้แก่ ชุดสัญลักษณ์มาตรฐานที่พัฒนาโดย Gane and Sarson (1979) และชุดสัญลักษณ์มาตรฐานที่พัฒนาโดย DeMarco and Yourdon (DeMarco, 1979; Yourdon and Constantine, 1979) โดยแต่ละชุดสัญลักษณ์มาตรฐานอธิบายโดยใช้สัญลักษณ์ 4 สัญลักษณ์ (กิตติและพนิดา, 2546: 150)



**รูปที่ 2** สัญลักษณ์และความหมายของ Yourdon และ Gane & Sarson

8.2.2 ระบบฐานข้อมูลและระบบจัดการฐานข้อมูล (Database System & Database Management System)

8.2.2.1 ระบบฐานข้อมูล (Database System)

ความหมายของระบบฐานข้อมูล โอภาส (2551: 35) กล่าวไว้ว่า ฐานข้อมูลคือศูนย์รวมของข้อมูลต่างๆ ที่มีความสัมพันธ์กัน (Relationship) โดยจะมีกระบวนการจัดหมวดหมู่ข้อมูลอย่างมีระเบียบแบบแผน ก่อให้เกิดฐานข้อมูลที่เป็นแหล่งรวมของข้อมูลจากแผนกต่างๆ ซึ่งถูกจัดเก็บไว้อย่างมีระบบภายในฐานข้อมูลชุดเดียวกัน โดยผู้ใช้งานแต่ละแผนกสามารถเข้าถึงข้อมูลส่วนกลางนี้เพื่อนําไปประมวลผลร่วมกัน ทําให้แก้ปัญหาความซ้ำซ้อนของข้อมูล และที่สําคัญข้อมูลในฐานข้อมูลจะไม่ผูกติดกับโปรแกรม กล่าวคือ จะมีความอิสระในข้อมูล (Program-Data Independence)

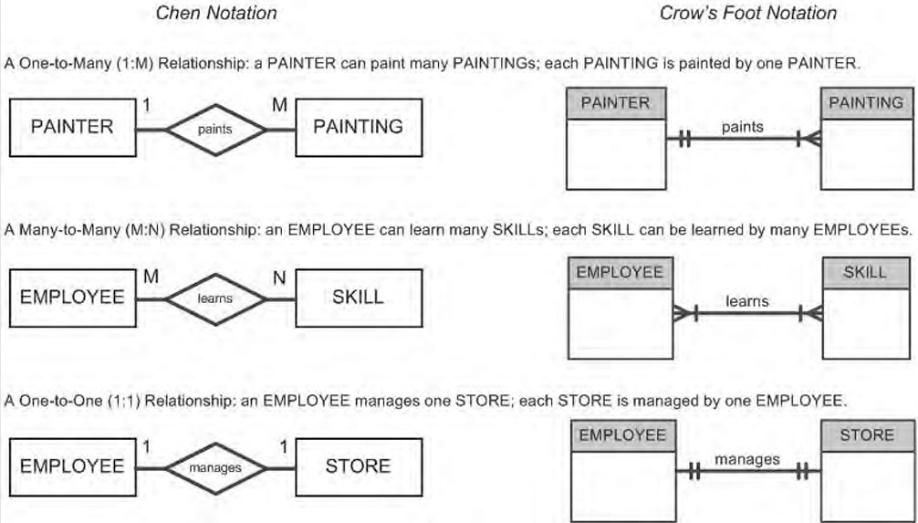
8.2.2.2 ระบบจัดการฐานข้อมูล (Database Management System : DBMS)

โอภาส (2551: 37-45) กล่าวว่า ระบบจัดการฐานข้อมูล มักเรียกย่อๆว่า DBMS คือซอฟต์แวร์ที่ใช้เป็นเครื่องมือของผู้ใช้เพื่อโต้ตอบกับฐานข้อมูล ซึ่งประกอบด้วยฟังก์ชันหน้าที่ต่างๆ ในการจัดการกับข้อมูล รวมทั้งภาษาที่ใช้ทํางานกับข้อมูล มักใช้กับ SQL ในการสร้าง, การเรียกดูและการบํารุงรักษาฐานข้อมูล นอกจากนี้ DBMS ยังมีหน้าที่ในการรักษาความมั่นคงและความปลอดภัยของข้อมูลด้วยการป้องกันมิให้ผู้ไม่มีสิทธิ์ใช้งานเข้ามาละเมิดข้อมูลในฐานข้อมูลที่เป็นศูนย์กลางได้รวมถึงการสํารองข้อมูลและการกู้คืนข้อมูล ในกรณีข้อมูลเกิดความเสียหาย เป็นต้น

8.2.2.3 แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (Entity Relationship Diagram : ER Diagram)

กิตติและพนิดา (2546: 200) ได้ให้ความหมายและข้อมูลที่เกี่ยวข้องกับแผนภาพแสดงความสัมพันธ์ระหว่างข้อมูลไว้ว่าการสร้างแผนภาพจําลองข้อมูลและกระบวนการดําเนินงานของระบบนั้นมีบทบาทสําคัญในการพัฒนาระบบ เนื่องจากสามารถแสดงโครงสร้างของข้อมูลและการทํางานภายในระบบชัดเจน ซึ่งจะช่วยให้ทั้งนักวิเคราะห์ระบบและผู้ใช้งานเกิดความเข้าใจในการทํางานของระบบอย่างถูกต้อง แบบจําลองข้อมูลที่สร้างขึ้นในขั้นตอนการวิเคราะห์ความต้องการของระบบนี้ยังเรียกว่าเป็น “การออกแบบฐานข้อมูลในระดับแนวความคิด (Conceptual Database Design)” ของขั้นตอนการออกแบบ (Design Phase) ในกิจกรรมการออกแบบฐานข้อมูลซึ่งจะนํา Conceptual Data Model ที่ได้จากกิจกรรมย่อยนี้ไปทําการปรับปรุงและออกแบบฐานข้อมูลในระดับ Logical และ Physical ต่อไป ในที่นี้เพื่อความสะดวกจะเรียก Conceptual Data Model ว่า “Data Model”แบบจําลองข้อมูล (Data Model) หมายถึง การจําลองข้อมูลที่เกิดขึ้นทั้งหมดในระบบพร้อมทั้งจําลองความสัมพันธ์ระหว่างข้อมูลที่เกิดขึ้นนั้นโดยใช้แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (Entity Relationship Diagram : ER Diagram) แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล หมายถึงแผนภาพที่ใช้เป็นเครื่องมือสําหรับจําลองข้อมูลซึ่งประกอบไปด้วย Entity (แทนกลุ่มของข้อมูลที่เป็นเรื่องเดียวกันหรือเกี่ยวข้องกัน) และความสัมพันธ์ระหว่างข้อมูล (Relationship) ที่เกิดขึ้นทั้งหมดในระบบ

สัญลักษณ์ที่ใช้ในแผนภาพ ER Diagram ที่ใช้ในการจําลองแบบข้อมูลมีหลายรูปแบบ แต่ที่นิยมใช้มีอยู่ 2 รูปแบบ คือ Chen Model และ Crow’s Foot Model (กิตติและพนิดา, 2546: 201-202)



**รูปที่ 3** สัญลักษณ์และความหมายของ Chen และ Crow’s Foot

8.3 งานวิจัยที่เกี่ยวข้อง

**9.วิธีการดำเนินโครงงาน**

**10.เครื่องมือที่ใช้พัฒนาระบบ**

**10.1 Visual Studio Code**

Visual Studio Code เป็นโปรแกรม Code Editor ที่พัฒนาโดยบริษัทไมโครซอฟท์ในรูปแบบ OpenSource ทำให้สามารถนำมาใช้งานได้ฟรี (Microsoft, n.d.) โปรแกรมนี้เหมาะสำหรับการพัฒนาโปรแกรมที่ต้องการใช้งานข้ามแพลตฟอร์ม โดยรองรับทั้งระบบปฏิบัติการ Windows, macOS และ Linux (White, 2020) สนับสนุนการใช้งานกับภาษาโปรแกรมต่าง ๆ เช่น JavaScript, TypeScript และ Node.js อีกทั้งยังสามารถเชื่อมต่อกับ Git ได้อย่างสะดวก และมีส่วนขยาย (Extensions) มากมายที่ช่วยเพิ่มความสามารถของโปรแกรม เช่น การเปิดใช้งานภาษา C++, C#, Java, Python เป็นต้น (Vasquez, 2019) มีจุดเด่นในด้านความเร็วในการโหลดและการใช้งานทรัพยากรน้อยที่สุด อีกทั้งยังสามารถปรับแต่งการใช้งานตามความต้องการของนักพัฒนาได้อย่างยืดหยุ่น (Stack Overflow, 2022) โดยมีความสามารถในการรองรับส่วนขยายที่หลากหลาย ทำให้เหมาะสมสำหรับนักพัฒนาทุกระดับ ตั้งแต่มือใหม่จนถึงมืออาชีพ

**10.2 MongoDB**

MongoDB เป็นฐานข้อมูลประเภท NoSQL ที่ได้รับความนิยมอย่างมากในปัจจุบัน โดยมีจุดเด่นอยู่ที่ความยืดหยุ่นในการจัดเก็บข้อมูลที่หลากหลายรูปแบบ โดยไม่จำเป็นต้องมีโครงสร้างที่ตายตัวเหมือนฐานข้อมูลแบบสัมพันธ์ (Relational Database) เช่น MySQL (MongoDB, n.d.; Membrey et al., 2014) ใช้การจัดเก็บข้อมูลในรูปแบบ JSON-like documents (BSON) ซึ่งช่วยให้การจัดการข้อมูลทำได้ง่ายและสะดวกยิ่งขึ้น โดยเหมาะสำหรับงานที่ต้องการรองรับข้อมูลที่มีการเปลี่ยนแปลงรูปแบบอย่างต่อเนื่อง และสามารถปรับขยายระบบเพื่อรองรับข้อมูลจำนวนมากได้โดยไม่ลดประสิทธิภาพ (Chodorow, 2013)

**10.3 Prisma**

Prisma เป็นเครื่องมือ ORM (Object-Relational Mapper) ที่ได้รับความนิยมอย่างมากในปัจจุบัน โดยถูกออกแบบมาเพื่อช่วยให้นักพัฒนาสามารถทำงานกับการเชื่อมต่อฐานข้อมูลได้ง่ายขึ้นและมีประสิทธิภาพมากขึ้น โดยเฉพาะอย่างยิ่งสำหรับภาษา JavaScript และ TypeScript (Prisma, n.d.; Bahdank, 2022) มีจุดเด่นที่รองรับการทำงานกับฐานข้อมูลหลากหลายประเภท เช่น PostgreSQL, MySQL, SQLite, และ MongoDB พร้อมด้วยอินเทอร์เฟซที่ใช้งานง่าย ช่วยให้การเขียน Query เป็นไปอย่างรวดเร็วและลดข้อผิดพลาดที่อาจเกิดขึ้น (Matsouka, 2021) นอกจากนี้ Prisma ยังรองรับการปรับแต่งโครงสร้างข้อมูลและการทำงานร่วมกับระบบ Type Safety ของ TypeScript ซึ่งช่วยเพิ่มความแม่นยำในการพัฒนาโปรแกรม (Wieruch, 2021)

**10.4 Next.js**

Next.js คือเฟรมเวิร์กที่สร้างขึ้นบน React โดยถูกออกแบบมาเพื่อช่วยให้นักพัฒนาสร้างเว็บแอปพลิเคชันได้อย่างรวดเร็วและมีประสิทธิภาพยิ่งขึ้น (Vercel, n.d.; Roth, 2021) หนึ่งในจุดเด่นของ Next.js คือความสามารถในการทำ Server-Side Rendering (SSR) และ Static Site Generation (SSG) ซึ่งช่วยให้เว็บไซต์โหลดเร็วขึ้นและเหมาะสำหรับการปรับปรุงประสิทธิภาพด้าน SEO เนื่องจากหน้าเว็บถูกสร้างขึ้นบนเซิร์ฟเวอร์ก่อนที่จะส่งไปยังเบราว์เซอร์ (Wieruch, 2022) ซึ่งมีฟีเจอร์ที่ช่วยลดความซับซ้อนในการพัฒนา เช่น ระบบการจัดการเส้นทาง (Routing) ที่ไม่ต้องตั้งค่าเพิ่มเติม และการทำงานร่วมกับ API ได้อย่างราบรื่น ทั้งนี้ Next.js เหมาะสำหรับโปรเจ็กต์ทุกขนาด ตั้งแต่เว็บไซต์เล็กๆ ไปจนถึงแอปพลิเคชันขนาดใหญ่ (Brito, 2021)

**10.5 Tailwind CSS**

Tailwind CSS คือ CSS Framework ประเภทหนึ่งที่เน้นการใช้งานแบบ utility-first โดยหมายถึงการใช้คลาสสำเร็จรูปที่ถูกออกแบบมาเพื่อควบคุมการจัดการสไตล์ต่าง ๆ โดยไม่จำเป็นต้องเขียน CSS เองตั้งแต่เริ่มต้น (Tailwind Labs, n.d.; Robinson, 2020) ตัวอย่างการใช้งานได้แก่การกำหนดสี, ขนาด, การจัดวาง, และเอฟเฟกต์ต่าง ๆ ซึ่งช่วยให้นักพัฒนาสามารถสร้างเลย์เอาต์และสไตล์ได้อย่างรวดเร็วและยืดหยุ่น ปรับแต่งได้ง่าย นักพัฒนาสามารถแก้ไขค่าต่าง ๆ เช่น สีและขนาดในไฟล์การตั้งค่า (Configuration File) เพื่อให้เหมาะสมกับโปรเจ็กต์ของตน นอกจากนี้ ยังช่วยลดปริมาณโค้ด CSS ที่ไม่จำเป็นผ่านการทำงานร่วมกับเครื่องมือ PurgeCSS ซึ่งช่วยลบคลาสที่ไม่ได้ใช้งานออกจากไฟล์ CSS สุดท้าย (Robinson, 2021) Tailwind CSS ได้รับความนิยมเพิ่มขึ้นเนื่องจากสามารถรวมเข้ากับเฟรมเวิร์กยอดนิยมอย่าง React, Vue, และ Next.js ได้อย่างราบรื่น และเหมาะสำหรับการพัฒนาเว็บไซต์ที่ต้องการความเร็วและความยืดหยุ่นสูง (Spencer, 2021)

**11. เอกสารอ้างอิง**

Boehm, C. (2007). **The discipline that never was: Current developments in music technology in higher education in Britain. Journal of Music**, Technology & Education, 1(1), 7–21.

Goodfellow, I., Bengio, Y., & Courville, A. (2014). **Deep learning**. MIT Press.

Herremans, D., Chuan, C.-H., & Chew, E. (2017). **A functional taxonomy of music generation systems.** ACM Computing Surveys, 50(5), 1–30.

Hracs, B. J., Jakob, D., & Hauge, A. (2013). **Standing out in the crowd: The rise of exclusivity-based strategies to compete in the contemporary marketplace for music and fashion.** Environment and Planning A, 45(5), 1144–1161.

Berkowitz, A. E. (2024). **Gimme Some Truth: AI Music and Implications for Copyright and Cataloging. Information Technology and Libraries,** 43(3). https://doi.org/10.5860/ital.v43i3.17072

MRArranger. (2023). **The Rise of AI in Music Production: Tools and Technologies.** Music Technology Review.

Nvidia. (2024). **Introducing Fugatto: AI-Powered Sound Creation.** Nvidia Blog.

Microsoft. (n.d.). **Visual Studio Code.** Retrieved from https://code.visualstudio.com

Fowler, M., & Parsons, M. (2018). **Refactoring: Improving the Design of Existing Code (2nd ed.).** Addison-Wesley.

White, T. (2020). M**astering Visual Studio Code: The Complete Guide to Using and Customizing VS Code for Developers.** O'Reilly Media.

Stack Overflow. (2022). **Developer Survey 2022.** Retrieved from https://insights.stackoverflow.com/survey/2022

Vasquez, C. (2019). **"Top extensions for Visual Studio Code".** Smashing Magazine. Retrieved from https://www.smashingmagazine.com

MongoDB. (n.d.). **What is MongoDB?.** Retrieved from https://www.mongodb.com

Membrey, P., Plugge, E., & Hawkins, T. (2014). **The Definitive Guide to MongoDB: A complete guide to dealing with Big Data using MongoDB.** Apress.

Chodorow, K. (2013). **MongoDB: The Definitive Guide: Powerful and Scalable Data Storage.** O'Reilly Media.

Prisma. (n.d.). **Next-generation Node.js and TypeScript ORM.** Retrieved from https://www.prisma.io

Bahdank, A. (2022). **Prisma ORM: A Practical Guide to Modern Database Management.** Dev.to. Retrieved from https://dev.to

Matsouka, A. (2021). **Building Scalable Applications with Prisma and TypeScript.** Medium. Retrieved from https://medium.com

Wieruch, R. (2021). **Prisma and TypeScript: Writing Safe Database Queries**. Retrieved from https://www.robinwieruch.de

Tailwind Labs. (n.d.). **Tailwind CSS Documentation.** Retrieved from https://tailwindcss.com

Robinson, A. (2020). **Mastering Tailwind CSS: Building Modern Websites and Web Applications**. O'Reilly Media.

Robinson, A. (2021). **Utility-First CSS Frameworks: Tailwind in Practice.** Smashing Magazine. Retrieved from https://www.smashingmagazine.com

Spencer, J. (2021). **Integrating Tailwind CSS with Modern JavaScript Frameworks**. Manning Publications.