

วิถีแห่ง SpecKit จากสเปกสู่โค้ด

ปลดล็อกประสิทธิภาพและความแม่นยำ
ในทุกขั้นตอนการพัฒนา

แองเจล่า เจริญ
อนุชิต ชลธรร

สำนักพิมพ์ก๊อปวาง

วิธีแห่ง SpecKit: จากสเปกสู่โค้ด

ปลดล็อกประสิทธิภาพและความแม่นยำ
ในทุกขั้นตอนการพัฒนา

แองเจล่า เหยียน

อนุชิต ชโลธร

คำนำสำนักพิมพ์

การเดินทางของเราในโลกของ **Spec-Driven Development** เริ่มต้นจากหนังสือ *Spec Driven Development* ที่เน้นวางรากฐานทางความคิดและทฤษฎีอย่างเจาะลึก แต่หลังจากหนังสือเล่มนั้นเผยแพร่ออกไป เสียงตอบรับจากผู้อ่านหลายท่าน โดยเฉพาะข้อเสนอแนะอันทรงคุณค่าชิ้นหนึ่ง ได้จุดประกายให้เราเห็นว่า ผู้อ่านต้องการหนังสือที่นำไป “ลงมือทำได้จริง” ไม่ใช่แค่ทฤษฎีบนหิ้ง

เราน้อมรับฟัง และนำคำแนะนำนั้นมาเป็นจุดเริ่มต้นของการสร้างสรรค์ครั้งใหม่

หนังสือ “วิถีแห่ง *SpecKit: จากสเปกสู่โค้ด*” เล่มนี้ คือผลลัพธ์ของการรับฟังและต่อยอด เราตั้งใจออกแบบให้เป็นคู่มือปฏิบัติ ที่จะพาคุณข้ามจากโลกของแนวคิดไปสู่การลงมือทำจริง ด้วยเครื่องมือที่จับต้องได้อย่าง **SpecKit** เครื่องมือที่จะเป็นสะพานเชื่อมระหว่าง “สเปก” และ “โค้ด” ทำให้การพัฒนาแบบ Spec-Driven กลายเป็นเรื่องที่คุณทำได้จริงในทุกวันทำงาน

เราขอขอบคุณผู้อ่านทุกท่านที่เป็นแรงบันดาลใจและมอบคำแนะนำให้เราเสมอมา หนังสือเล่มนี้เกิดขึ้นได้เพราะเสียงของพวกคุณ และเราหวังว่ามันจะเป็นเพื่อนร่วมทางที่ช่วยให้คุณสร้างสรรค์ซอฟต์แวร์ที่ดี

กว่าเดิม พร้อมเดินไปบนเส้นทางการพัฒนาอย่างมั่นคงและมีเป้าหมายที่ชัดเจน

คำนำผู้เขียน

ในยุคที่ซอฟต์แวร์ซับซ้อนขึ้นทุกวัน ช่องว่างระหว่าง “สิ่งที่เราคิดจะสร้าง” กับ “สิ่งที่เราสร้างขึ้นมาจริงๆ” กลายเป็นปัญหาใหญ่ที่สุดอย่างหนึ่งของการพัฒนาซอฟต์แวร์

หลายทีมพยายามใช้เอกสารอย่าง PRD, ADR หรือ Confluence Note เพื่อบันทึกรายละเอียดของระบบ แต่สุดท้ายเอกสารเหล่านั้นก็มักจะ “ล้าสมัย” กว่าซอร์สโค้ดเสมอ ความไม่ตรงกันนี้เองที่นำไปสู่บั๊ก ความเข้าใจที่คลาดเคลื่อน และการสูญเสียเวลาอย่างน่าเสียดาย

SpecKit ถือกำเนิดขึ้นเพื่อแก้ปัญหานี้โดยเฉพาะ มันไม่ใช่แค่เครื่องมือ แต่คือ “กระบวนทัศน์ใหม่” ในการพัฒนาที่เรียกว่า **Spec-Driven Development (SDD)** ซึ่งมีหัวใจสำคัญคือ “สเปก” (Specification)

ในโลกของ SDD สเปกไม่ใช่แค่เอกสารอธิบายระบบ แต่เป็น “**สัญญา ระหว่างมนุษย์และเครื่องจักร**” เป็นจุดศูนย์กลางของการสื่อสารระหว่างทีมพัฒนา, AI Agent และเครื่องมืออัตโนมัติต่างๆ ที่ทำงานร่วมกัน

SpecKit จึงเป็นสะพานที่เชื่อมระหว่าง “แนวคิด” กับ “การลงมือทำ” ตั้งแต่การนิยามระบบ การวางแผน ไปจนถึงการเขียนโค้ด โดยมีเครื่องมือ CLI ที่ชื่อว่า **specify** เป็นศูนย์กลางของกระบวนการ

ทั้งหมด ไม่ว่าคุณจะเป็นนักพัฒนา ผู้จัดการโครงการ หรือสถาปนิกซอฟต์แวร์ SpecKit จะช่วยให้ทุกคนมองภาพเดียวกัน ผ่านภาษาที่ทั้งคนและเครื่องจักรเข้าใจตรงกัน

หนังสือเล่มนี้จะพาคุณไปรู้จัก SpecKit ตั้งแต่พื้นฐานจนถึงการประยุกต์ใช้จริง เริ่มตั้งแต่การติดตั้ง specify การสร้าง “ธรรมนูญ” (Constitution) ของโครงการ การออกแบบสเปก ไปจนถึงการนำ AI Agent เข้ามาช่วยในแต่ละขั้นตอน นอกจากนี้ยังมีตัวอย่างโครงร่างสเปกและแนวคิดเบื้องหลังที่คุณสามารถนำไปปรับใช้กับทีมของคุณได้ทันที

เราหวังว่าหนังสือเล่มนี้จะไม่เพียงสอนให้คุณ “ใช้ SpecKit เป็น” แต่จะช่วยให้คุณ “คิดแบบ Spec-Driven” ได้ นั่นคือการมองซอฟต์แวร์เป็นระบบที่สามารถอธิบาย ทำความเข้าใจ และพัฒนาได้อย่างมีแบบแผนและสื่อสารได้ชัดเจนยิ่งขึ้น

เพราะ SpecKit ไม่ได้เปลี่ยนแค่เครื่องมือที่คุณใช้ แต่จะเปลี่ยนวิธีที่คุณและทีมคิดในการพัฒนาซอฟต์แวร์ไปตลอดกาล

สารบัญ

บทที่ 1: รู้จักกับ SpecKit	1
ทำไมต้องใช้ SpecKit?	1
Spec-Driven Development คืออะไร?	2
ปัญหาที่ SpecKit เข้ามาแก้	3
ภาพรวมกระบวนการพัฒนาด้วย SpecKit	4
สถาปัตยกรรมและส่วนประกอบหลัก	5
บทที่ 2: ปรัชญาและแนวคิดเบื้องหลัง SpecKit	7
ปรัชญาของ SpecKit: 3 เสาหลักสู่การสร้างซอฟต์แวร์แนวใหม่	7
SpecKit กับแนวทางปฏิบัติอื่นๆ	8
เทียบกับ Product Requirements Document (PRD)	9
เทียบกับ Architecture Decision Record (ADR)	9
เทียบกับ C4 Model	9
การเดินทางของ SpecKit: บทเรียนและเป้าหมาย	10
บทที่ 3: เริ่มต้นใช้งาน	12
การติดตั้ง Specify CLI	12
การติดตั้งแบบถาวร (Permanent Installation)	12
การใช้งานแบบชั่วคราว (Temporary Execution)	13
ตรวจสอบความพร้อมของระบบ	14

สร้างโปรเจกต์แรกของคุณ	15
เจาะลึกคำสั่ง Specify CLI	16
คำสั่งหลัก	16
ตัวเลือกที่ใช้บ่อยของ specify init	17
ตัวอย่างการใช้งาน	17
โครงสร้างโปรเจกต์ SpecKit	18
หัวใจของโปรเจกต์: ไฟล์ constitution.md	18
สร้าง “ธรรมนูญ” ของโปรเจกต์	19
ตัวอย่างไฟล์ธรรมนูญ	20
บทที่ 4: การสร้างสเปก	23
หลักการออกแบบสเปกที่ดี	23
สร้างสเปกฉบับร่างด้วย AI Agent	24
จัดเกลาสเปกให้เฉียบคม	25
ความสัมพันธ์ระหว่าง “สเปก” และ “ธรรมนูญ”	26
ตัวอย่าง spec.md	26
บทที่ 5: การวางแผนและจัดการงาน	32
สร้างแผนเชิงเทคนิค (Technical Plan)	32
ตัวอย่าง plan.md	33
แปลงสเปกเป็นรายการงาน (Tasks)	36
ตัวอย่าง tasks.md	37
วงจรการทำงานร่วมกันระหว่างคนกับ AI	40

บทที่ 6: การลงมือพัฒนาจากสเปก	42
Workflow การพัฒนาที่ขับเคลื่อนด้วย AI	42
ตัวอย่าง: การใช้ AI Agent พัฒนา API	43
การทดสอบคือหัวใจสำคัญ	44
วงจรการปรับปรุง	44
บทที่ 7: การทวนสอบและตรวจสอบคุณภาพ	46
สร้าง Checklist อัตโนมัติด้วย /speckit.checklist	46
วิเคราะห์โค้ดเทียบกับสเปกด้วย /speckit.analyze	47
บทที่ 8: กระบวนการทำงานครบวงจร	49
ภาพรวม Workflow ของ SpecKit	49
วางรากฐานของโปรเจกต์	50
วงจรการพัฒนาฟีเจอร์	51
การทวนสอบและรับฟีดแบ็ก	52
ฟีดแบ็กคือหัวใจของความยืดหยุ่น	53
บทที่ 9: กรณีศึกษา พลิกโฉม Legacy Code ด้วย SpecKit	54
บริบท: การ Refactor ระบบ E-commerce รุ่นเก่า	54
สภาพก่อนใช้ SpecKit (The “Before” State)	54
ภารกิจ: Refactor ระบบโปรโมชัน	56
กระบวนการพลิกโฉมด้วย SpecKit	56

สภาพหลังใช้ SpecKit	59
บทสรุปจากกรณีศึกษา	61
บทที่ 10: การทำงานร่วมกับเครื่องมืออื่น	63
สเปกคือ API ของทีม	63
เชื่อมต่อกับเครื่องมือจัดการโปรเจกต์ (Jira, Asana, GitHub Issues)	64
ตัวอย่าง: สร้าง GitHub Issues จากไฟล์ tasks.md	64
ผนวกเข้ากับ CI/CD Pipeline	66
ตัวอย่าง: ตรวจสอบความครอบคลุมของเทสต์เทียบกับสเปกใน GitHub Actions	66
สร้างเอกสารอัตโนมัติ (Automated Documentation)	68
ตัวอย่าง: สร้าง API Reference (OpenAPI/Swagger) จากสเปก	68
บทที่ 11: การใช้ AI Agents ใน SpecKit	71
AI Agents ที่ SpecKit รองรับ	71
แนวคิด “Bring Your Own Agent” (BYOA)	72
Workshop: การเชื่อมต่อกับ Agent ที่ไม่รองรับโดยตรง	73
การใช้ AI ช่วยในแต่ละเฟสของ SpecKit	75
บทที่ 12: คู่มืออ้างอิงคำสั่ง	78
รูปแบบคำสั่งมาตรฐาน	78
กลุ่มคำสั่งสำหรับ Specification	78

/speckit.constitution	79
/speckit.specify	79
/speckit.clarify	80
กลุ่มคำสั่งสำหรับ Planning	80
/speckit.plan	81
/speckit.tasks	81
กลุ่มคำสั่งสำหรับ Implementation & Verification	81
/speckit.implement	81
/speckit.checklist	82
/speckit.analyze	82
การสร้าง Custom Commands	83
ตัวอย่าง: สร้างคำสั่ง /speckit.docgen	83
Best Practices ในการใช้คำสั่ง	84
บทที่ 13: เทคนิคขั้นสูง	86
การบริหารจัดการ Constitution ขั้นสูง	86
ใครคือผู้ดูแลไฟล์ธรรมนูญ?	86
การจัดการเวอร์ชันของไฟล์ธรรมนูญ	87
วิธีที่ดีและวิธีที่ควรหลีกเลี่ยงในการเขียนสเปก	88
วิธีที่ดี	88
วิธีที่ควรหลีกเลี่ยง	89
เจาะลึกการสร้าง Custom Commands	90

ตัวอย่าง: สร้างคำสั่ง /speckit.refactor	90
บทที่ 14: การแก้ปัญหาและคำถามที่พบบ่อย	93
การแก้ปัญหา	93
ปัญหาเกี่ยวกับการติดตั้ง	93
ปัญหาเกี่ยวกับ AI Agent	94
คำถามที่พบบ่อย	96
ตัวอย่าง: สร้างเว็บไซต์บริษัทด้วย Next.js และ shadcn/ui	99
วางรากฐานของโปรเจกต์	99
สร้างข้อกำหนดของพีเจอร	100
จัดความคลุมเครือ	102
จากไอเดียสู่สเปกที่จับต้องได้	103
ตัวอย่าง: สร้าง API สำหรับค้นหาสินค้า ด้วย Node.js	105
วางรากฐานของโปรเจกต์	105
นิยามพีเจอรให้คมชัด	105
User Scenarios	106
Technical Requirements	106
วางแผนการพัฒนอย่างเป็นระบบ	107
Tech Stack & Project Setup	107
Project Structure	107
Generated Artifacts	108

แปลงแผนสู่รายการงานที่ชัดเจน	108
เปลี่ยนสเปกให้เป็นโค้ด	109
ผลลัพธ์สุดท้าย	110
ตัวอย่าง: การสร้างเว็บไซต์จากงานออกแบบ	112
การสร้างงานออกแบบด้วย Stich	112
วางรากฐานของโปรเจกต์	115
สร้างข้อกำหนดของพีเจอร์	116
วางแผนการพัฒนาอย่างเป็นระบบ	119
แปลงแผนสู่รายการงานที่ชัดเจน	119
เปลี่ยนสเปกให้เป็นโค้ด	120
จากไอเดียเป็นสเปกที่นำไปพัฒนาต่อได้จริง	120
บทส่งท้าย	121
สรุปการเดินทางกับ SpecKit	121
อนาคตของ Spec-Driven Development	122
ก้าวต่อไปของคุณ	123
SpecKit ไม่ใช่ยาวิเศษ	123
ดาวน์โหลดซอร์สโค้ด	125

บทที่ 1: รู้จักกับ SpecKit

ยินดีต้อนรับสู่โลกของการพัฒนาซอฟต์แวร์ที่ขับเคลื่อนด้วยสเปก หรือ **Spec-Driven Development (SDD)** บทนี้จะพาคุณไปทำความรู้จักกับ **SpecKit** ว่ามันคืออะไร ถูกสร้างขึ้นมาเพื่อแก้ปัญหาใด และแนวคิดเบื้องหลังของมันจะเข้ามาเปลี่ยนวิธีที่ทีมของคุณสร้างซอฟต์แวร์ไปตลอดกาลได้อย่างไร

ทำไมต้องใช้ SpecKit?

SpecKit ไม่ได้เป็นเพียง “เครื่องมือ” ขึ้นหนึ่งเท่านั้น แต่ถูกออกแบบขึ้นมาเพื่อแก้ “ปัญหาใหญ่” ที่ทีมพัฒนาซอฟต์แวร์ทั่วโลกต้องเจอเหมือนกันเสมอ นั่นคือ ช่องว่างระหว่างความคิด เอกสาร และโค้ด ซึ่งมักไม่สอดคล้องกันและเดินสวนทางอยู่บ่อยครั้ง

ลองนึกภาพการทำงานตามปกติ

1. **Product Manager** เขียนเอกสาร PRD อธิบายฟีเจอร์ใหม่
2. **Software Architect** ออกแบบระบบ และจัดทำเอกสาร ADR
3. **Developer** อ่านเอกสารทั้งหมด แล้วลงมือเขียนโค้ด
4. **QA Tester** กลับไปอ่านเอกสารเหล่านั้นอีกครั้ง เพื่อสร้าง Test Case