AI AGENT CO-WORKING SYSTEM · DAY 1

First Vibe Coding

สร้างระบบ CRM สโมสรแบบพื้นฐาน

จากไอเดียสู่ระบบ CRM ที่ล็อกอินได้จริง — เข้าใจ Frontend · Backend · Database และทำงานกับทีม AI Agent อย่างถูกวิธี



STEP 01
ออกแบบ
Use Case · Permission Matrix · SPEC.md

STEP 02
สร้าง
Main Agent สร้างทีละชั้น ฐานข้อมูล หลังบ้าน หน้าบ้าน

STEP 03
ล็อกอิน
3 บทบาท ล็อกสิทธิ์ถึงฐานข้อมูล

STEP 04
ขึ้นเว็บจริง
Cloudflare Pages เปิดบนมือถือได้

GETTING STARTED

วิธีใช้คู่มือเล่มนี้

คู่มือเรียงตามลำดับของวันเรียน ตั้งแต่เตรียมเครื่อง เข้าใจระบบ 3 ชั้น ออกแบบก่อนสร้าง สร้าง CRM ด้วยทีม Subagent ทำระบบล็อกอิน 3 บทบาท จนขึ้นเว็บจริง ทุกขั้นมีคำอธิบายสั้น ๆ และ Prompt สำเร็จรูปที่คัดลอกไปวางใน AI Agent (Claude Code หรือ Codex) ได้ทันที

ปุ่มคัดลอก
ในคู่มือฉบับเว็บ ทุกกล่อง Prompt มีปุ่ม "คัดลอก Prompt" มุมขวาบน

[ ] ตัวแดงในวงเล็บ
เปลี่ยนเป็นข้อมูลของตัวเองก่อนส่ง

ด่านทดสอบ
สิ่งที่ต้องเห็นบนจอจริงก่อนไปขั้นถัดไป

กล่องแดงอ่อน
หลักคิดหรือข้อควรระวังที่สำคัญ

LEARNING OUTCOMES

ผลลัพธ์หลังเรียน

จบวันนี้ ทีมสโมสร (เจ้าของ/ผู้จัดการ · แอดมินและดูแลสมาชิก · ฝ่ายขายและการตลาด) จะเข้าใจว่าระบบ 1 ระบบทำงานอย่างไร และได้ระบบ CRM ต้นแบบที่ใช้งานได้จริงกลับไป — ไม่ต้องมีพื้นฐานเขียนโปรแกรม

- ✓ **อธิบายได้** ว่า 1 ระบบประกอบด้วย Frontend · Backend · Database และข้อมูลไหลอย่างไร
- ✓ **วาง Use Case ได้** ว่าระบบมีกี่บทบาท แต่ละบทบาททำอะไรได้ + เขียนตารางสิทธิ์ (Permission Matrix)
- ✓ **เข้าใจการทำงานจริงของ AI Agent / Subagent** เลือกใช้ให้ถูกงาน และเขียน SPEC.md + ไฟล์กติกาทีมให้ Agent อ่าน
- ✓ **ได้ระบบ CRM ต้นแบบ:** ฟอร์มสนใจสมัคร → ลีดเข้าฐานข้อมูล → หลังบ้านจัดการลีด + ล็อกอิน 3 บทบาท
- ✓ **ขึ้นเว็บจริง** บน Cloudflare Pages เปิดบนมือถือได้

ชิ้นงานที่ได้ระหว่างวัน

`Use Case Diagram` `Permission Matrix` `SPEC.md` `CLAUDE.md / AGENTS.md` `ระบบ CRM บน Cloudflare Pages`

DAY OVERVIEW

ตารางสอน

1 วัน 10:00–17:00 น. · พักกลางวัน 12:30–13:30 น. · เบรก 15:00–15:15 น.

เวลา	ช่วง	หัวข้อ	ผลลัพธ์ / ด่าน
10:00–10:30	**Session 1**	Vibe Coding คืออะไร + เปิดโจทย์สโมสร	แผนที่โจทย์: วันนี้ทำอะไร อะไรต่อยอด
10:30–11:00	**Session 2**	ระบบ 1 ระบบประกอบด้วยอะไร (Frontend / Backend / Database + Authentication / Authorization)	อธิบายระบบสโมสรเป็น 3 ชั้นได้
11:00–11:30	**Session 3**	AI Agent ทำงานจริงอย่างไร (Main Agent · Subagent · ไฟล์กติกา)	เข้าใจว่าทำไมต้องเขียน SPEC และไฟล์กติกาให้ดี
11:30–12:30	**Workshop 1**	ออกแบบก่อนสร้าง: Use Case · หน้าจอตามบทบาท · Permission Matrix · SPEC.md · ไฟล์กติกา	Use Case Diagram · Permission Matrix · SPEC.md · CLAUDE.md / AGENTS.md
12:30–13:30	พักกลางวัน		
13:30–15:00	**Workshop 2**	สร้าง CRM ด้วย Main Agent ทีละชั้น (Database → Backend → Frontend)	**ด่าน 1** กรอกฟอร์มแล้วลีดโผล่ใน CRM · เปลี่ยนสถานะแล้วค่าในตารางเปลี่ยน
15:00–15:15	เบรก		
15:15–16:30	**Workshop 3**	ระบบล็อกอินและบทบาท (แอดมิน / พนักงานขาย / ผู้บริหาร) + ล็อกสิทธิ์ที่ฐานข้อมูล	**ด่าน 2** สลับล็อกอิน 3 บทบาท เห็น/ทำได้ตรง Permission Matrix
16:30–17:00	**Workshop 4**	ขึ้นเว็บจริง + สรุป + แผนต่อยอด	**ด่าน 3** เปิด URL บนมือถือแล้วล็อกอินได้

> **หมายเหตุถึงผู้เรียน**
> ตารางเวลาเป็นแนวทาง ผู้สอนอาจปรับเวลาแต่ละช่วงให้เหมาะกับความคืบหน้าของผู้เรียน โดยเนื้อหาและผลลัพธ์หลักยังครบถ้วน

BEFORE CLASS

ก่อนวันเรียน — เตรียมเครื่อง

เตรียมบัญชีและเครื่องให้พร้อมก่อนวันเรียน แล้วเริ่มลงมือได้ทันทีตั้งแต่ Session 1

- ✓ **คอมพิวเตอร์ + เบราว์เซอร์ Chrome**
- ✓ **บัญชี Claude (Pro ขึ้นไป) หรือ ChatGPT (Plus ขึ้นไป)**
- ✓ **ติดตั้ง Claude Code หรือ Codex**
- ✓ **สมัคร Supabase (ฟรี) และ Cloudflare (ฟรี)**
  ใช้อีเมลที่ใช้ประจำ

> **ไม่ต้องติดตั้งอะไรเพิ่มเอง**
> Setup 1 ให้ Agent เช็คและติดตั้ง Git / Node.js ให้

เครื่องมือในคลาส

AI Agent
Claude Code หรือ Codex — ใช้ได้ทั้งสองแอป

Supabase
ฐานข้อมูล + ระบบล็อกอิน

Cloudflare Pages
หน้าเว็บ + Backend ผ่าน Pages Functions

Git
ตั้งจุดย้อนกลับของงาน

Stack ของระบบในคลาส

HTML/CSS/JavaScript ธรรมดา ไม่มี build step · supabase-js จาก CDN · Backend = Cloudflare Pages Functions (`functions/api/…`) · Database = Supabase Postgres + Auth + Row Level Security

CLAUDE CODE หรือ CODEX

วิธีเรียกใช้ต่างแอป

Prompt ทุกตัวในคู่มือเขียนแบบกลาง ใช้ได้ทั้งสองแอป เช่น "ไฟล์บันทึกโปรเจกต์ (CLAUDE.md / AGENTS.md)" และ "ถามฉันเป็นตัวเลือกให้กดเลือก หรือให้ตอบสั้น ๆ ทีละชุด แล้วรอคำตอบ" — ความต่างมีเท่านี้

เรื่อง	Claude Code	Codex
ไฟล์บันทึกโปรเจกต์ (คู่มือกติกาทีม)	`CLAUDE.md`	`AGENTS.md`
สร้างระบบในคลาส	Main Agent ตัวเดียว ทำทีละชั้น	เหมือนกัน — ใช้ Prompt เดียวกัน
การถามคำถาม	ตัวเลือกให้กดเลือก	ฟอร์มข้อ ๆ ในแชท
แนบไฟล์/รูป	ลากไฟล์เข้าแชท หรือพิมพ์ path	ลากไฟล์เข้าแชท หรือพิมพ์ path

---

SETUP · ทำครั้งเดียว ช่วง SESSION 1

Setup 1–3 — เตรียมเครื่องมือและโปรเจกต์

ทำครั้งเดียวในช่วง Session 1 ได้เครื่องที่พร้อม โปรเจกต์ Supabase และโฟลเดอร์โปรเจกต์ที่มี Git

Setup 1 เช็คเครื่องมือ

ให้ Agent เช็ค Git, Node.js, npx และติดตั้งตัวที่ขาด

PROMPT S1 — Setup 1 · เช็คเครื่องมือในเครื่อง

```
ช่วยเช็คเครื่องของฉันว่าพร้อมทำเว็บหรือยัง:
- ตรวจว่ามี Git, Node.js (เวอร์ชัน 20 ขึ้นไป) และ npx หรือยัง
- ตัวไหนยังไม่มี ให้บอกวิธีติดตั้งที่ง่ายที่สุดสำหรับเครื่องนี้ แล้วช่วยติดตั้งให้ (ถามก่อนทุกครั้งที่ต้องใช้รหัสผ่านเครื่อง)
สรุปผลเป็นตาราง: เครื่องมือ · เวอร์ชัน · พร้อมใช้ / ต้องติดตั้ง
```

Setup 2 สร้างโปรเจกต์ Supabase (ทำเองบนเว็บ)

1. เข้า Supabase แล้วกด **New project**
2. ตั้งชื่อโปรเจกต์ `club-crm`
3. ตั้งรหัสผ่านฐานข้อมูล แล้วบันทึกเก็บไว้
4. เลือก Region **Singapore**
5. รอ 1–2 นาทีให้โปรเจกต์พร้อม
6. ไปที่ **Project Settings → API** คัดลอก **Project URL** และ **anon / publishable key** เก็บไว้ใช้ใน Workshop 2

anon / publishable key
ใส่หน้าเว็บได้

service_role / secret key
= กุญแจห้องเก็บของ **ห้ามใส่หน้าเว็บเด็ดขาด**

Setup 3 ตั้งโฟลเดอร์โปรเจกต์ + Git

PROMPT S3

Setup 3 · ตั้งโฟลเดอร์โปรเจกต์ + Git

```
ช่วยตั้งโปรเจกต์ใหม่ชื่อ club-crm ให้ฉัน (ระบบ CRM ของสโมสรสุขภาพ ใช้ข้อมูลจำลองเท่านั้น):
- โครงโฟลเดอร์: public/ (หน้าเว็บ) · functions/api/ (Backend บน Cloudflare Pages Functions) · supabase/ (ไฟล์ SQL) · docs/ (เอกสารออกแบบ)
- เริ่ม Git และสร้าง .gitignore กันไฟล์ลับ (.dev.vars, .env) และ node_modules ไม่ให้หลุดขึ้น Git
- ยังไม่ต้องเขียนหน้าเว็บ แค่โครงพร้อมใช้
อธิบายสั้น ๆ ว่าแต่ละโฟลเดอร์คือชั้นไหนของระบบ (Frontend / Backend / Database)
```

SESSION 1 · 10:00–10:30

S1 Vibe Coding คืออะไร + เปิดโจทย์สโมสร

Vibe Coding = บอกสิ่งที่ต้องการเป็นภาษาคน ให้ AI Agent เขียนโค้ด แล้วเราทดสอบ ตรวจ และตัดสินใจ — เราเป็น "เจ้าของงาน" ไม่ใช่คนพิมพ์โค้ด



วงจรสร้างระบบ

01	02	03	04	05
เขียน SPEC	สร้าง	ทดสอบ	ขึ้นเว็บ	ดูแล

โจทย์สโมสร: 10 ความต้องการ

ทุกโมดูลต้องเสียบเข้า **"ฐานข้อมูลกลาง + CRM"** จึงเริ่มจากแกนนี้



10 ความต้องการ เสียบเข้าฐานข้อมูลกลาง + CRM

`Chatbot` `Customer Service` `Marketing` `Content` `Sales & CRM` `Document & Contract` `Booking` `Membership & Waiting List` `Website & Join Now` `Central Management`

วันนี้ทำอะไร อะไรต่อยอด

ระดับ	โมดูล
ทำวันนี้	Sales & CRM
ทำบางส่วน	Central Management (ฐานข้อมูลกลาง สิทธิ์ตามบทบาท) · Website & Join Now (เฉพาะฟอร์มลงทะเบียน)
ต่อยอด	Membership & Waiting List · Booking · Document & Contract · Chatbot / Customer Service
นอกขอบเขต	Marketing / Content Creator (งานผลิตสื่อ)

กติกา 3 ข้อของห้องเรียน

ใช้ข้อมูลจำลองในห้อง

ข้อมูลทั้งหมดในคลาสเป็นข้อมูลจำลอง · สโมสรตัวอย่างชื่อสมมติ **Harmony Club**

คีย์ลับไม่อยู่บนหน้าเว็บ

คีย์ลับอยู่ฝั่ง Backend เท่านั้น

คนอนุมัติเรื่องสำคัญ

AI ร่างได้ คนตัดสินใจ

SESSION 2 · 10:30–11:00

S2 ระบบ 1 ระบบประกอบด้วยอะไร

เปรียบระบบเหมือนร้านอาหาร 3 ส่วนที่ทำงานต่อกัน



ร้านอาหาร = Frontend / Backend / Database

Frontend (หน้าบ้าน)

= หน้าร้าน/เมนู/พนักงานรับออเดอร์ — สิ่งที่ลูกค้าเห็นและกด (หน้าฟอร์มสมัคร หน้ารายการลีด)

Backend (หลังบ้าน)

= ครัว — รับออเดอร์ ตรวจ ทำตามกฎ แล้วส่งของ (รับข้อมูลฟอร์ม ตรวจความถูกต้อง สร้างลีด เก็บคีย์ลับ)

Database (ฐานข้อมูล)

= คลังวัตถุดิบ/ตู้เก็บของ — เก็บข้อมูลถาวรเป็นตาราง (ลีด ประวัติการติดต่อ ทีมงาน)

ตามรอย 1 คำขอ "ลูกค้ากดสมัคร"



ตามรอยคำขอ "ลูกค้ากดสมัคร"

1. ลูกค้ากรอกฟอร์มบนหน้าเว็บ **(Frontend)**
2. ส่งไป `/api/join` **(Backend)** ตรวจข้อมูล + ใช้คีย์ลับ
3. บันทึกแถวใหม่ในตาราง `leads` **(Database)**
4. ตอบกลับ "ได้รับข้อมูลแล้ว"
5. ทีมงานเปิด CRM เห็นลีดใหม่

ล็อกอินทำงานอย่างไร



บัตรพนักงาน: Authentication vs Authorization

Authentication

คุณคือใคร

แตะบัตรพนักงานที่ประตู

Authorization

คุณทำอะไรได้

บัตรแต่ละใบเปิดได้แค่บางห้อง

คีย์ลับอยู่ชั้นไหน

อยู่ **Backend เท่านั้น** — Frontend ใครก็เปิดดูโค้ดได้

Mini-activity · ให้ Agent อธิบายระบบสโมสรเป็น 3 ชั้น

`PROMPT S2-EXPLAIN`

Session 2 · ให้ Agent อธิบายระบบเป็น 3 ชั้น

```
ฉันกำลังจะทำระบบ CRM ให้สโมสรสุขภาพ: ลูกค้ากรอกฟอร์มสนใจสมัคร → ทีมขายติดตาม → ผู้บริหารดูภาพรวม
อธิบายให้คนที่ไม่เคยเขียนโปรแกรมเข้าใจว่า ระบบนี้แบ่งเป็น Frontend / Backend / Database อย่างไร
- แต่ละชั้นทำหน้าที่อะไร เปรียบกับร้านอาหาร
- ตามรอย 1 คำขอ "ลูกค้ากดส่งฟอร์มสมัคร" ว่าผ่านชั้นไหนบ้างตามลำดับ
- คีย์ลับของฐานข้อมูลควรอยู่ชั้นไหน เพราะอะไร
ตอบสั้น กระชับ เป็นข้อ ๆ ยังไม่ต้องเขียนโค้ด
```

SESSION 3 · 11:00–11:30

S3 AI Agent ทำงานจริงอย่างไร

✕ **ความเชื่อผิด**

"Agent หลายตัวคุยกันเอง นัดแบ่งงานกันเอง ทำงานอัตโนมัติไปเรื่อย ๆ"

✓ **ความจริง**

"Agent ไม่ได้คุยกันเอง แต่ทำงานผ่านใบสั่งงาน กติกากลาง และข้อมูลที่ใช้ร่วมกัน"

Main Agent กับ Subagent ทำงานกัน 5 ขั้น



หัวหน้า (Main Agent) แจกใบสั่งงาน Frontend / Backend / Database

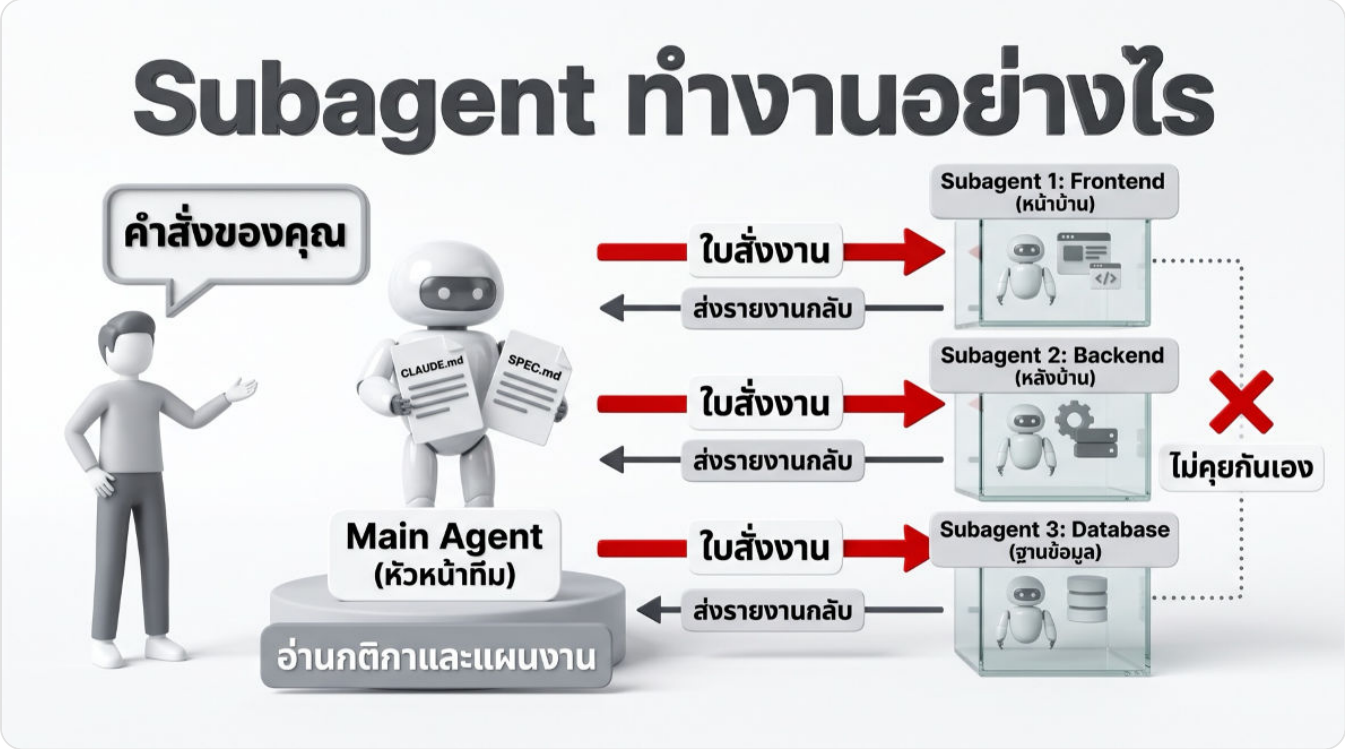
1. **Main Agent (หัวหน้าทีม)**
   คุยกับเรา อ่านโจทย์ แล้วตัดสินใจว่าจะแตกงานไหม

2. **หัวหน้าเขียนใบสั่งงาน (brief)**
   ให้ Subagent แต่ละตัว — Frontend / Backend / Database

3. **Subagent เริ่มจากความจำว่าง**
   ไม่เห็นบทสนทนาของเรา รู้แค่: ใบสั่งงาน + ไฟล์กติกา CLAUDE.md / AGENTS.md + ไฟล์ในโปรเจกต์ (SPEC.md, โค้ด)

4. **ทำเสร็จส่งรายงานกลับหัวหน้าเท่านั้น**
   — Subagent ด้วยกันไม่คุยกัน ถ้าต้องรู้งานของกันและกันจะรู้ผ่านไฟล์ที่ใช้ร่วมกัน

5. **หัวหน้ารวมผล ตรวจ แล้วรายงานเรา**



Subagent ทำงานอย่างไร

หมายเหตุความถูกต้อง

มีระบบแบบทีม (เช่น agent teams) ที่ส่งข้อความหากันได้ แต่เป็นช่องทางที่ออกแบบไว้ และเริ่มจากคำสั่งของคนหรือหัวหน้าเสมอ — ไม่ใช่ AI ตื่นมาประชุมกันเอง

บทเรียน

SPEC.md และไฟล์กติกาที่ดี = ใบสั่งงานที่ดี = งานที่ถูกต้อง

ทำไมคลาสนี้สร้างด้วย Main Agent ตัวเดียว ไม่แตก Subagent

Subagent ช่วยให้เร็ว ไม่ได้ช่วยให้ถูก

งานเดฟที่ทุกชั้นพึ่งกัน ให้ Main Agent ตัวเดียวทำทีละชั้น แล้วทดสอบทีละชั้น

1. **ทุกชั้นพึ่งกัน**
   Frontend เรียก API ของ Backend · Backend ใช้ตารางของ Database — แยกกันทำ ต่างคนต่างตัดสินใจ (ชื่อคอลัมน์ รูปแบบข้อมูล) รวมแล้วไม่เข้ากัน
2. **โทเค็นแพงขึ้น**
   ระบบหลาย Agent ใช้โทเค็นราว 15 เท่าของการแชตปกติ (Agent ตัวเดียวราว 4 เท่า) เพราะทุกตัวต้องอ่านบริบทซ้ำ
3. **ที่แพงจริงคือรอบแก้ ไม่ใช่ความเร็ว**
   ทำพร้อมกันเสร็จเร็วขึ้นแต่ไม่ถูกลง ถ้าชิ้นงานไม่เข้ากัน ทุกรอบแก้ต้องอ่านโค้ดใหม่ทั้งก้อน
4. **เริ่มจากแบบที่ง่ายที่สุด**
   แล้วค่อยเพิ่มความซับซ้อนเมื่อจำเป็นเท่านั้น

Subagent เหมาะกับ	ไม่เหมาะกับ
ค้นหา / อ่านโค้ดแล้วสรุป · รีวิวหรือตรวจทาน (อ่านอย่างเดียว) · งานที่แยกขาดกันจริง	เขียนหลายชั้นของระบบเดียวพร้อมกัน · งานที่ต้องคุยโต้ตอบบ่อย · งานเล็กแก้จุดเดียว

อ้างอิง Anthropic · Multi-agent research system · Anthropic · Building effective agents · Cognition · Don't Build Multi-Agents · Claude Code Docs · Subagents

Agent 2 แบบ

	Agent ช่วยสร้างระบบ	Agent ที่ทำงานในระบบ
ตัวอย่าง	Claude Code / Codex	แชตบอตตอบลูกค้า, ผู้ช่วยสรุปลีด
ทำงานเมื่อไหร่	เมื่อคนสั่งในเครื่อง	ตลอด 24 ชม. บนเว็บที่ขึ้นแล้ว
อะไรสั่งให้เริ่ม	คนพิมพ์คำสั่ง	เหตุการณ์ (trigger) เช่น ลีดใหม่เข้า ลูกค้าทักแชต ถึงเวลาที่ตั้งไว้
ทำงานร่วมกันผ่าน	ใบสั่งงาน + CLAUDE.md / AGENTS.md + ไฟล์	ฐานข้อมูลกลาง + กฎ Workflow

"อัตโนมัติ" ในระบบจริง

01 **เกิดเหตุการณ์**

02 **ระบบเรียก AI ตามกฎ**

03 **เขียนผลลงฐานข้อมูลกลาง**

04 **ส่วนอื่นอ่านไปทำต่อ**

05 **เรื่องสำคัญรอคนกดอนุมัติ**

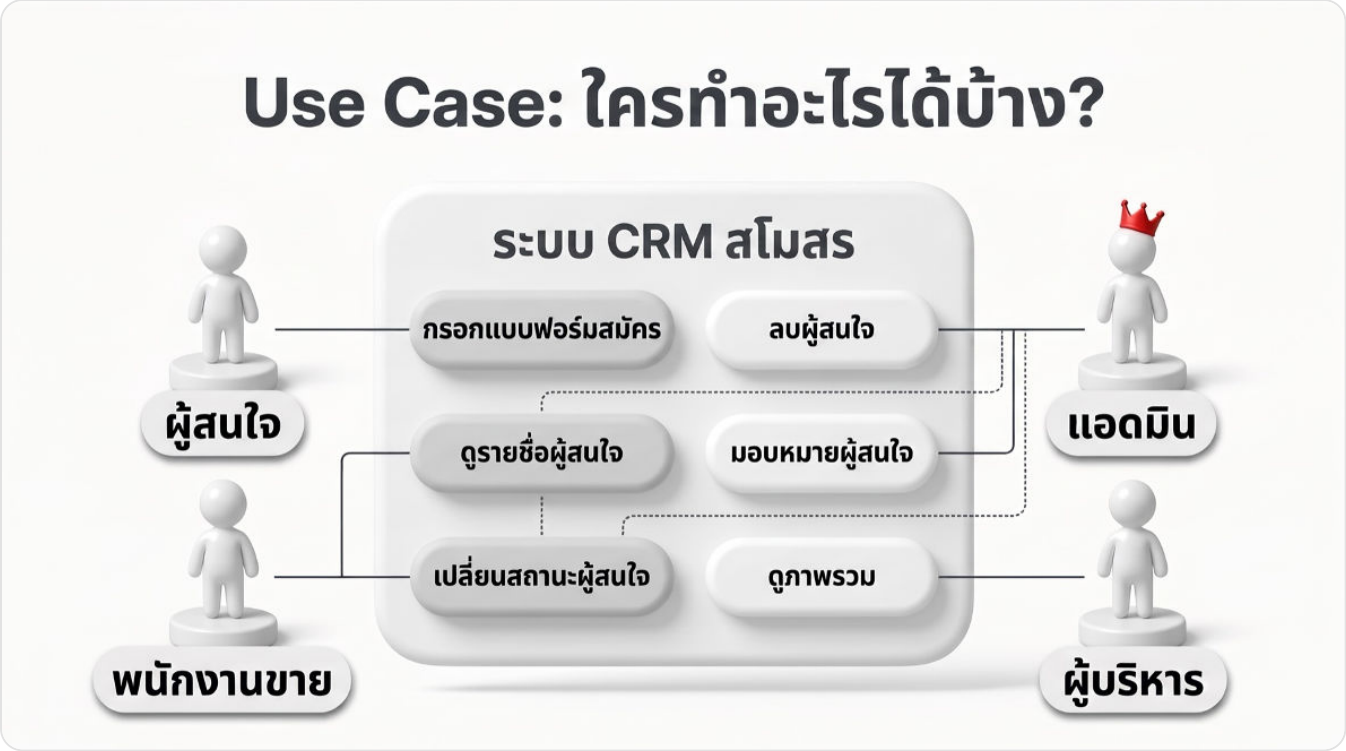
ตัวอย่าง: ลีดใหม่ → AI สรุปความสนใจ + ร่างข้อความ → แจกลีดให้ฝ่ายขาย → ฝ่ายขายกดส่งเอง — เป็นงานต่อยอด (ดูแผนต่อยอดข้อ 7)

WORKSHOP 1 · 11:30–12:30

W1 ออกแบบก่อนสร้าง

เป้าหมาย: ได้แบบระบบครบก่อนสั่งสร้าง — Use Case Diagram · Permission Matrix · ข้อมูล 3 ตาราง · SPEC.md · ไฟล์กติกา

1.1 Use Case — 4 บทบาท



4 บทบาทรอบระบบ

บทบาท	ทำได้	ทำไม่ได้
ผู้สนใจ (ไม่ต้องล็อกอิน)	ดูประเภทสมาชิกและสิทธิ์ · กรอกฟอร์มสนใจสมัคร · เลือกให้/ไม่ให้ความยินยอมการตลาด	เห็นข้อมูลลูกค้าคนอื่น · เข้าหลังบ้าน
แอดมิน	เห็นลีดทุกคน ค้นหา กรอง · เพิ่ม/แก้/ลบลีด · มอบหมายลีดให้พนักงานขาย · เพิ่มทีมงานและกำหนดบทบาท · จัดการประเภทสมาชิก	—
พนักงานขาย (ผู้ใช้ทั่วไป)	ดูลีดที่ได้รับมอบหมาย · เปลี่ยนสถานะตามขั้นการขาย · บันทึกการโทร/แชต/นัดหมาย	ลบลีด · เพิ่มทีมงานหรือเปลี่ยนบทบาท
ผู้บริหาร	ดูตัวเลขสรุป (ลีด อัตราสมัคร งานค้าง) · ดูลีดตามช่องทางและสถานะ · ดูรายชื่อลีดทั้งหมด	แก้ไขหรือลบข้อมูล (ดูอย่างเดียว)

PROMPT W1-USECASE — Workshop 1.1 · วาด Use Case Diagram

```
ระบบ CRM ของสโมสรสุขภาพ Harmony Club (ชื่อสมมติ) มีผู้ใช้ 4 บทบาท:
1. ผู้สนใจ (ไม่ต้องล็อกอิน): ดูประเภทสมาชิก · กรอกฟอร์มสนใจสมัคร · เลือกให้/ไม่ให้ความยินยอมด้านการตลาด
2. แอดมิน: เห็นลีดทุกคน ค้นหา กรอง · เพิ่ม/แก้/ลบลีด · มอบหมายลีดให้พนักงานขาย · เพิ่มทีมงานและกำหนดบทบาท · จัดการประเภทสมาชิก
3. พนักงานขาย: ดูเฉพาะลีดที่ได้รับมอบหมาย · เปลี่ยนสถานะ · บันทึกการโทร/แชต/นัดหมาย · ลบไม่ได้
4. ผู้บริหาร: ดูตัวเลขสรุปและรายชื่อลีดทั้งหมด · แก้ไขไม่ได้
วาด Use Case Diagram ด้วย Mermaid แล้วบันทึกที่ docs/use-case.md
ใต้แผนภาพ ทำตาราง "บทบาท · ทำได้ · ทำไม่ได้" และบอกว่ามีบทบาทหรือการกระทำไหนที่ฉันลืมคิด (ยังไม่ต้องเพิ่ม ให้ถามก่อน)
```

1.2 หน้าจอตามบทบาท

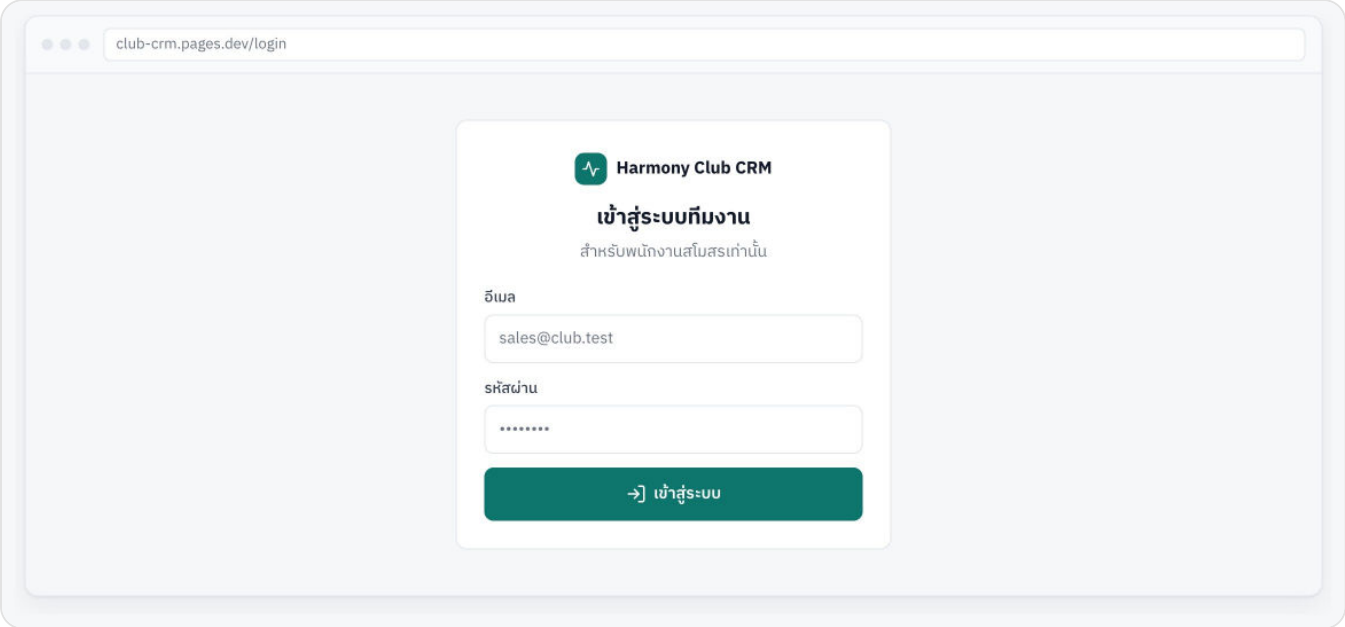
ดูหน้าจอตัวอย่าง (mockups/roles.html) ก่อนเขียนตารางสิทธิ์ — ระบบเดียวกัน ฐานข้อมูลเดียวกัน แต่หน้าจอ เมนู และปุ่มที่แต่ละคนเห็นไม่เท่ากัน ขึ้นกับว่าล็อกอินเข้ามาในบทบาทไหน

4 ขั้นหลังกดล็อกอิน

01	02	03	04
กรอกอีเมล + รหัสผ่าน	ระบบตรวจว่าเป็นใคร (Authentication)	อ่านบทบาทจากตาราง staff	เปิดเมนูตามสิทธิ์ (Authorization)

หน้าล็อกอิน

`/login`



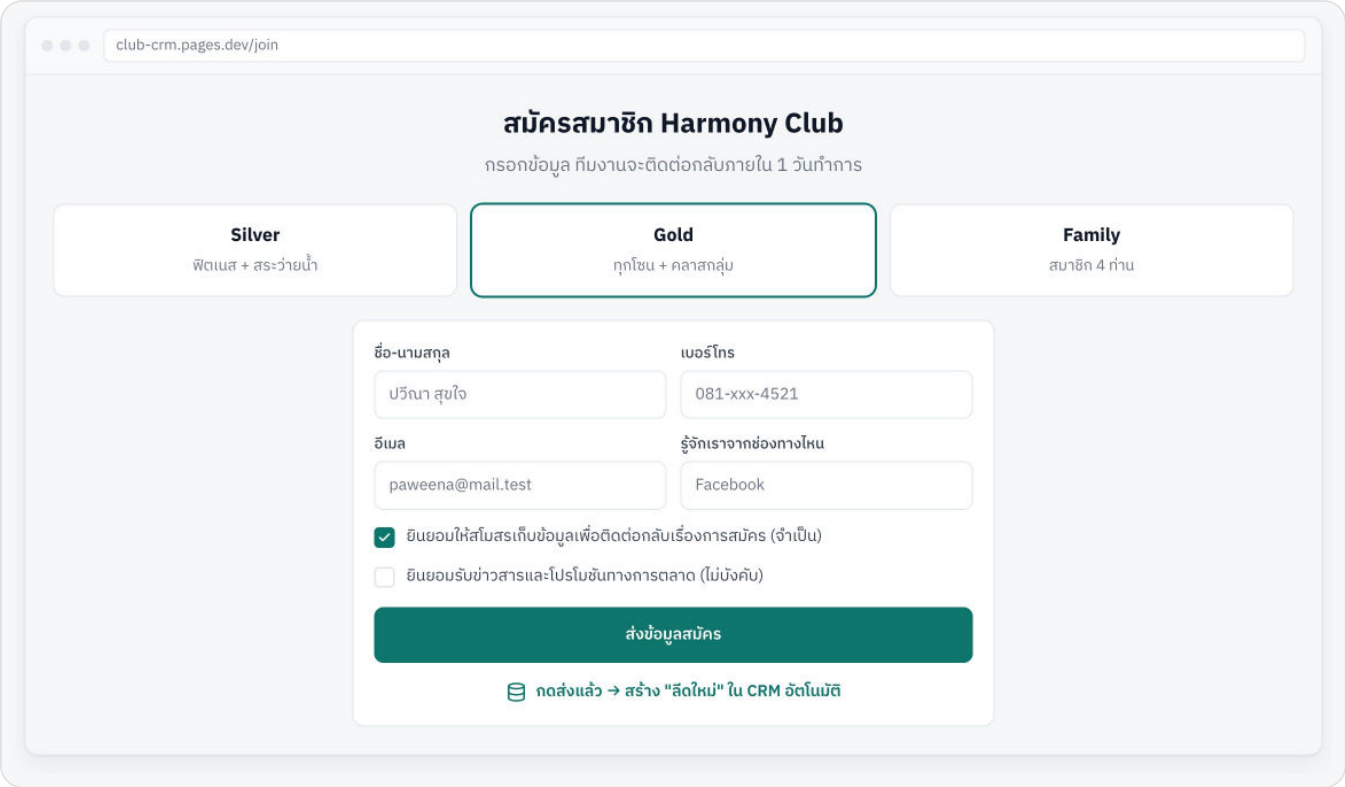
หน้าล็อกอิน — ทุกบทบาทเข้าทางประตูเดียวกัน

บัญชีทดสอบ (ข้อมูลจำลอง)

แอดมิน `admin@club.test` · พนักงานขาย `sales@club.test` · ผู้บริหาร `manager@club.test` — สลับเข้าทีละบทบาทเพื่อเทียบหน้าจอ

ผู้สนใจสมัครสมาชิก

ไม่ต้องล็อกอิน



ฟอร์มสนใจสมัคร — กดส่งแล้วสร้าง "ลีดใหม่" ใน CRM อัตโนมัติ

ทำได้

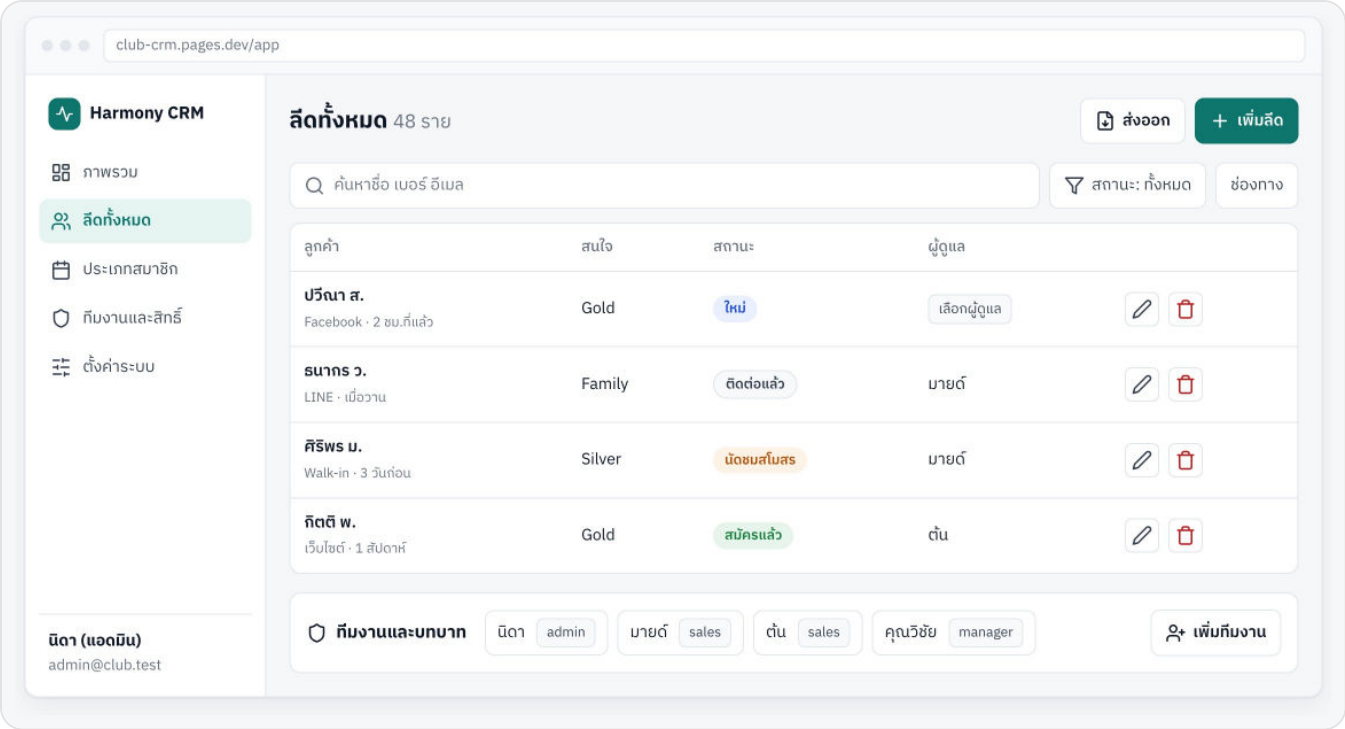
ดูประเภทสมาชิกและสิทธิ์ · กรอกฟอร์มสนใจสมัคร · เลือกให้/ไม่ให้ความยินยอมด้านการตลาด

ทำไม่ได้

มองไม่เห็นข้อมูลลูกค้าคนอื่น และเข้าหลังบ้านไม่ได้

แอดมิน

`role: admin`



แอดมิน — ผู้ดูแลระบบและฐานข้อมูลสมาชิก สิทธิ์สูงสุด

ทำได้

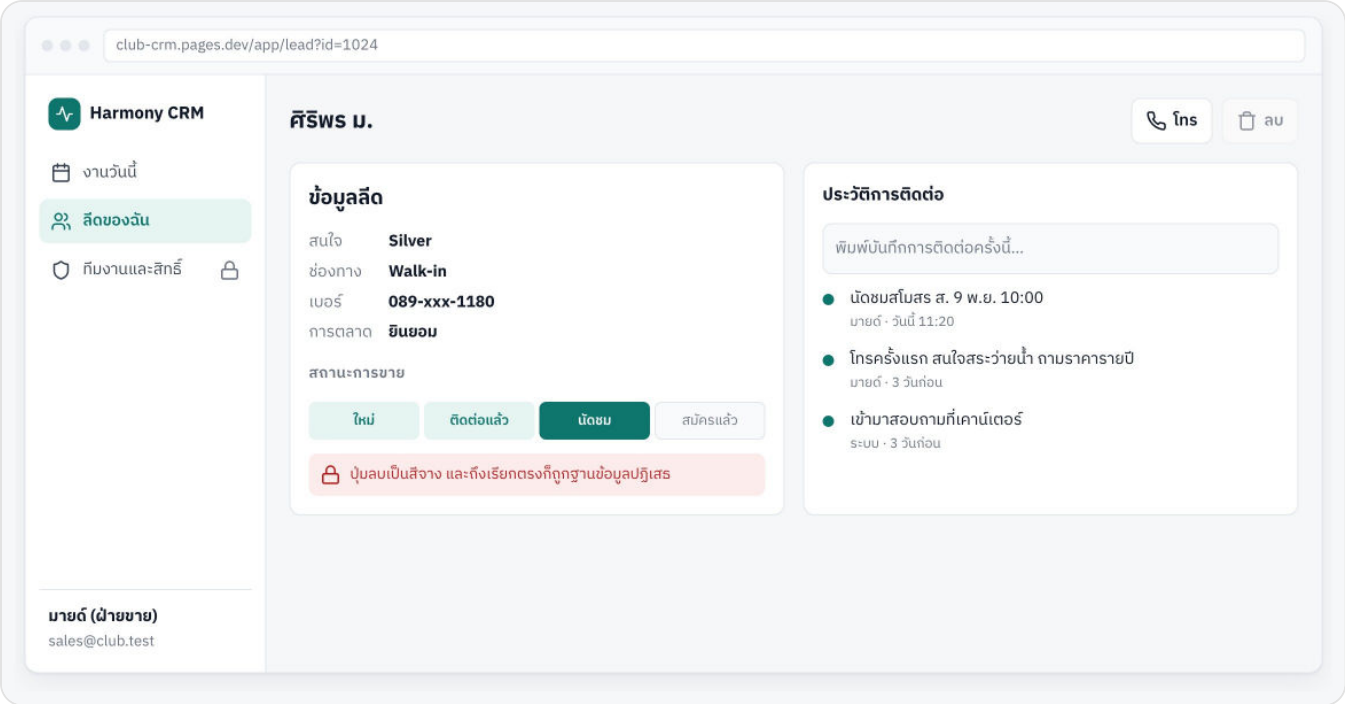
เห็นลีดทุกคน ค้นหาและกรอง · เพิ่ม แก้ไข และลบลีด · มอบหมายลีดให้พนักงานขาย · เพิ่มบัญชีทีมงานและกำหนดบทบาท · จัดการประเภทสมาชิก

ทำไม่ได้

— (สิทธิ์สูงสุด)

พนักงานขาย / ผู้ใช้ทั่วไป

`role: sales`



พนักงานขาย — ปุ่มลบเป็นสีจาง และถึงเรียกตรงก็ถูกฐานข้อมูลปฏิเสธ

ทำได้

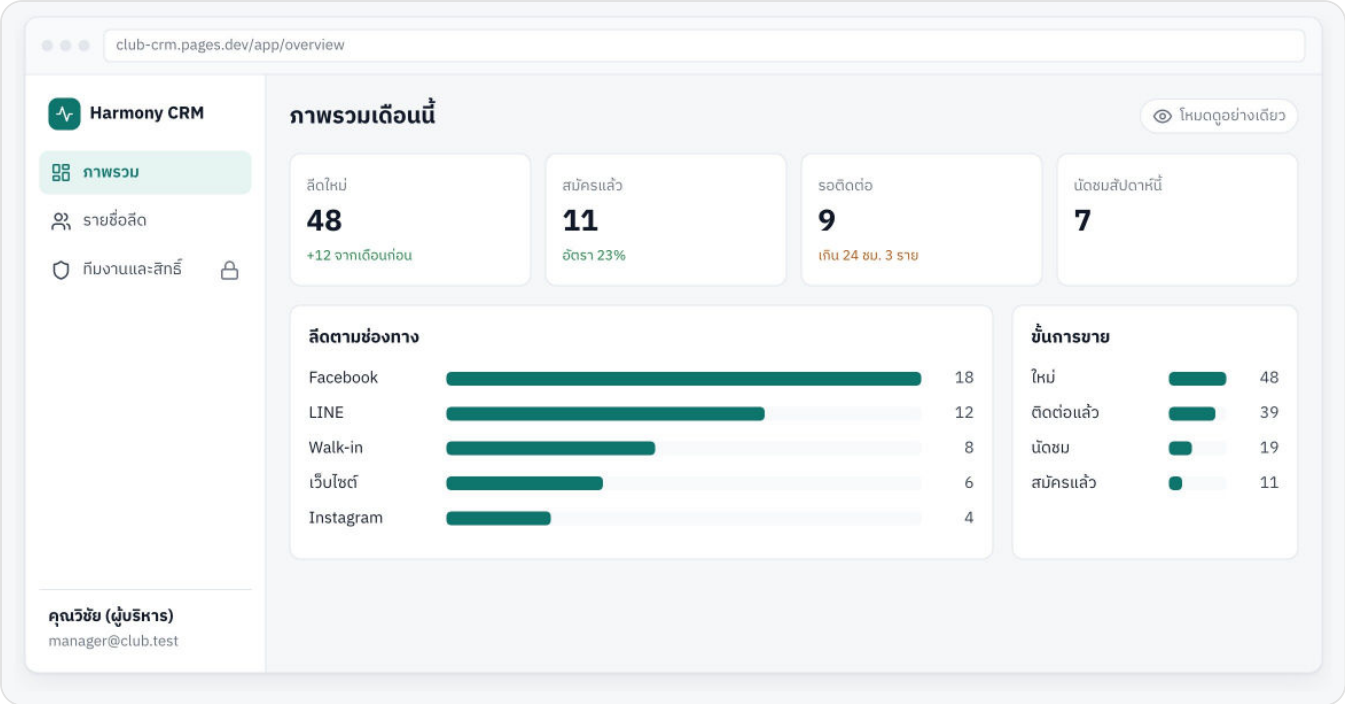
ดูลีดที่ได้รับมอบหมาย · เปลี่ยนสถานะตามขั้นการขาย · บันทึกการโทร/แชต/นัดหมาย

ทำไม่ได้

ลบลีด · เพิ่มทีมงานหรือเปลี่ยนบทบาท

ผู้บริหาร

`role: manager`



ผู้บริหาร — เจ้าของ / ผู้จัดการสโมสร ดูภาพรวมเพื่อตัดสินใจ

ทำได้

ดูตัวเลขสรุป: ลีด อัตราสมัคร งานค้าง · ดูลีดแยกตามช่องทางและสถานะ · ดูรายชื่อลีดทั้งหมด

ทำไม่ได้

แก้ไขหรือลบข้อมูล (ดูอย่างเดียว)

1.3 Permission Matrix — ตารางสิทธิ์

การกระทำ	ผู้สนใจ	แอดมิน	พนักงานขาย	ผู้บริหาร
กรอกฟอร์มสนใจสมัคร	ได้	ได้	ได้	ได้
ดูรายชื่อลีด	ไม่ได้	ได้	เฉพาะของตัวเอง	ได้
เปลี่ยนสถานะ / บันทึกการติดต่อ	ไม่ได้	ได้	ได้	ไม่ได้
ลบลีด	ไม่ได้	ได้	ไม่ได้	ไม่ได้
มอบหมายลีดให้พนักงาน	ไม่ได้	ได้	ไม่ได้	ไม่ได้
ดูภาพรวมตัวเลข	ไม่ได้	ได้	ไม่ได้	ได้
เพิ่มทีมงาน / กำหนดบทบาท	ไม่ได้	ได้	ไม่ได้	ไม่ได้

ตารางนี้คือสิ่งที่เขียนลง SPEC.md ก่อนสั่ง Agent สร้าง แล้วใช้เป็นด่านตรวจตอนจบ Workshop 3

PROMPT W1-MATRIX

```
จาก docs/use-case.md ทำตารางสิทธิ์ (Permission Matrix) บันทึกที่ docs/permissions.md
- แถว = การกระทำ (กรอกฟอร์มสนใจสมัคร · ดูรายชื่อลีด · เปลี่ยนสถานะ/บันทึกการติดต่อ · ลบลีด · มอบหมายลีด · ดูภาพรวมตัวเลข · เพิ่มทีมงาน/กำหนดบทบาท)
- คอลัมน์ = ผู้สนใจ · แอดมิน · พนักงานขาย · ผู้บริหาร
- ช่อง = ได้ / ไม่ได้ / ได้บางส่วน (ระบุเงื่อนไข เช่น "เฉพาะลีดของตัวเอง")
ตารางนี้จะใช้เป็นด่านทดสอบตอนท้าย Workshop 3 ให้เขียนแบบที่ทดสอบได้จริง
```

1.4 ข้อมูล 3 ตาราง

ให้ AI เสนอ แล้วคนตรวจก่อนอนุมัติ



ตู้เก็บข้อมูล 3 ตาราง

ทีมงาน

`staff`

`id` (ผูกกับบัญชีล็อกอิน) · `full_name` · `role` (admin / sales / manager)

ลีด

`leads`

`id` · `full_name` · `phone` · `email` · `interest_plan` (Silver / Gold / Family) · `channel` (Facebook / LINE / Walk-in / เว็บไซต์ / Instagram) · `status` · `assigned_to` (→ staff) · `consent_contact` · `consent_marketing` · `created_at` · `updated_at`

ประวัติการติดต่อ

`contact_logs`

`id` · `lead_id` (→ leads) · `staff_id` (→ staff) · `note` · `created_at`

สถานะลีด



เส้นทางสถานะลีด

`new` ใหม่	`contacted` ติดต่อแล้ว	`tour` นัดชมสโมสร	`member` สมัครแล้ว	`lost` ไม่สนใจ

ใหม่ → ติดต่อแล้ว → นัดชมสโมสร → สมัครแล้ว / ไม่สนใจ

PROMPT W1-DATA — Workshop 1.4 · ออกแบบข้อมูล 3 ตาราง (ยังไม่สร้าง)

```
ออกแบบฐานข้อมูลสำหรับ CRM นี้บน Supabase (Postgres) — เสนอให้ดูก่อน ยังไม่ต้องสร้างจริง
ใช้ 3 ตาราง:
- staff: ทีมงาน ผูกกับบัญชีล็อกอินของ Supabase Auth · ชื่อ · role (admin / sales / manager)
- leads: ลีด · ชื่อ · เบอร์ · อีเมล · ประเภทที่สนใจ (Silver / Gold / Family) · ช่องทางที่รู้จัก · สถานะ (new /
contacted / tour / member / lost) · ผู้ดูแล (→ staff) · ยินยอมให้ติดต่อ · ยินยอมการตลาด · เวลาสร้าง/แก้ไข
- contact_logs: ประวัติการติดต่อ · ลีด (→ leads) · พนักงาน (→ staff) · บันทึก · เวลา
แสดงเป็นตาราง: ชื่อคอลัมน์ · ชนิดข้อมูล · ตัวอย่างค่า · ความหมายภาษาไทย
แล้ววาดความสัมพันธ์ด้วย Mermaid (erDiagram) บันทึกที่ docs/data.md
ถ้าเห็นว่าควรเพิ่มหรือตัดอะไร ให้ถามฉันเป็นตัวเลือกให้กดเลือก หรือให้ตอบสั้น ๆ ทีละชุด แล้วรอคำตอบ
```

1.5 SPEC.md

รวมทุกอย่างข้างบน + Stack + หน้าจอ + ด่านทดสอบ ไว้ในไฟล์เดียว

PROMPT W1-SPEC

Workshop 1.5 · เขียน SPEC.md

```
รวมทุกอย่างที่เราตกลงกันเขียนเป็น SPEC.md ที่ root ของโปรเจกต์ ให้ Agent ตัวไหนมาอ่านก็สร้างตามได้:
1. เป้าหมายระบบ (CRM สโมสร เวอร์ชันพื้นฐาน · ข้อมูลจำลองเท่านั้น)
2. Stack: HTML/CSS/JavaScript ธรรมดา ไม่มี build step · supabase-js จาก CDN · Backend = Cloudflare Pages Functions · Database = Supabase (Postgres + Auth + Row Level Security)
3. บทบาทและตารางสิทธิ์ (อ้าง docs/use-case.md และ docs/permissions.md)
4. ข้อมูล 3 ตาราง (อ้าง docs/data.md)
5. หน้าจอ: /join ฟอร์มสนใจสมัคร · /login · /app รายการลีด · /app/lead หน้ารายละเอียดลีด · /app/overview ภาพรวมผู้บริหาร · /app/team จัดการทีมงาน (แอดมิน)
6. แบ่งงานตาม 3 ชั้น: Database (supabase/) · Backend (functions/api/) · Frontend (public/) — ระบุไฟล์ของแต่ละชั้นให้ชัด และสร้างทีละชั้นตามลำดับนี้ ทดสอบทุกชั้นก่อนไปต่อ
7. ด่านทดสอบ: ด่าน 1 ฟอร์ม → ลีดเข้า CRM · ด่าน 2 สลับล็อกอิน 3 บทบาทตรงตารางสิทธิ์ · ด่าน 3 เปิดบนมือถือได้
8. สิ่งที่ยังไม่ทำ (ต่อยอดภายหลัง): แบบประเมิน % · Audit Log · Waiting List · ระบบจอง · สัญญา · แชตบอต
```

1.6 ไฟล์กติกาทีม CLAUDE.md / AGENTS.md

"คู่มือที่ Agent ทุกตัวต้องอ่าน"

PROMPT W1-RULES

Workshop 1.6 · เขียนไฟล์กติกาทีม (CLAUDE.md / AGENTS.md)

```
สร้างไฟล์บันทึกโปรเจกต์ (CLAUDE.md / AGENTS.md) ให้เป็น "คู่มือกติกาทีม" ที่ Agent และ Subagent ทุกตัวต้องอ่านก่อนทำงาน ใส่:
- โปรเจกต์นี้คืออะไร + ให้อ่าน SPEC.md ก่อนเริ่มงานทุกครั้ง
- กติกาข้อมูล: ใช้ข้อมูลจำลองเท่านั้น · ห้ามใส่ service_role key หรือคีย์ลับใด ๆ ในโฟลเดอร์ public/ · คีย์ลับอยู่ใน .dev.vars และ Cloudflare secret เท่านั้น
- กติกาสิทธิ์: ทุกตารางเปิด Row Level Security · สิทธิ์ต้องตรงกับ docs/permissions.md
- กติกาการทำงาน: ทำทีละชั้น แก้เฉพาะไฟล์ของชั้นที่กำลังทำ · ทำเสร็จให้รายงานว่าแก้ไฟล์อะไร ทดสอบอะไร แล้วรอให้ฉันทดสอบ · ไม่แน่ใจให้ถาม ห้ามเดา
- ภาษา: อธิบายผู้ใช้เป็นภาษาไทย ข้อความบนหน้าเว็บเป็นภาษาไทย
เขียนให้สั้น อ่านจบใน 1 นาที
```

WORKSHOP 2 · 13:30–15:00

W2 สร้าง CRM ด้วย Main Agent ทีละชั้น

เป้าหมาย: ระบบ CRM เวอร์ชันแรกรันบนเครื่อง — ฟอร์มสนใจสมัครส่งลีดเข้าฐานข้อมูล + หลังบ้านจัดการลีด

2.1 เชื่อม Supabase

ไฟล์	เก็บอะไร	เปิดเผยได้ไหม
`public/config.js`	Project URL + anon key	เปิดเผยได้
`.dev.vars`	service_role key	**ไฟล์ลับ ไม่ขึ้น Git**

PROMPT W2-CONNECT

Workshop 2.1 · เชื่อมโปรเจกต์กับ Supabase

```
ช่วยตั้งค่าเชื่อมโปรเจกต์กับ Supabase:
- สร้าง public/config.js เก็บ SUPABASE_URL และ SUPABASE_ANON_KEY (คีย์นี้เปิดเผยได้)
- สร้าง .dev.vars เก็บ SUPABASE_URL และ SUPABASE_SERVICE_ROLE_KEY (คีย์ลับ ใช้เฉพาะ Backend) และยืนยันว่า
.dev.vars อยู่ใน .gitignore
- ทำช่องว่างไว้ แล้วบอกฉันว่าต้องไปคัดลอกค่าไหนจากหน้าไหนใน Supabase มาวางเอง (ฉันจะวางคีย์เองในไฟล์ ไม่ส่งในแชต)
```

2.2 Main Agent สร้างทีละชั้น

Main Agent อ่าน SPEC.md แล้วสร้างเองทีละชั้น หยุดให้เราทดสอบทุกชั้นก่อนไปต่อ · เหตุผลที่ไม่แตก Subagent ดู Session 3 · ใช้ได้เหมือนกันทั้ง Claude Code และ Codex

ชั้น 1 · Database (ฐานข้อมูล)

`supabase/schema.sql` สร้าง 3 ตาราง + `supabase/seed.sql` ข้อมูลจำลอง 20 ราย

ทดสอบ: เห็นตารางใน Table Editor (ข้อ 2.3)

ชั้น 2 · Backend (หลังบ้าน)

`functions/api/join.js` รับข้อมูลฟอร์ม ตรวจความถูกต้อง สร้างลีดสถานะ new ด้วยคีย์ลับ

ทดสอบ: ส่งข้อมูลตัวอย่างแล้วลีดเข้า

ชั้น 3 · Frontend (หน้าบ้าน)

หน้า `/join` ฟอร์มสนใจสมัคร + หน้า `/app` รายการลีด ค้นหา กรอง + หน้ารายละเอียดลีด

ทดสอบ: ด่าน 1

PROMPT W2-BUILD

Workshop 2.2 · Main Agent สร้างทีละชั้น (Database → Backend → Frontend)

```
อ่าน SPEC.md และไฟล์บันทึกโปรเจกต์ แล้วสร้าง CRM เวอร์ชันแรกด้วยตัวเองทีละชั้นตามลำดับ (ไม่ต้องแตก subagent)
ทำเสร็จแต่ละชั้นให้หยุด สรุปสั้น ๆ ว่าแก้ไฟล์อะไร แล้วบอกวิธีให้ฉันทดสอบ รอฉันยืนยันก่อนไปชั้นถัดไป
1. Database (ฐานข้อมูล) — แก้เฉพาะ supabase/
   schema.sql สร้าง 3 ตาราง staff / leads / contact_logs ตาม docs/data.md และเปิด Row Level Security
ทุกตาราง (ช่วงนี้อนุญาตให้ผู้ที่ล็อกอินอ่าน/แก้ได้ก่อน จะล็อกตามบทบาทใน Workshop 3)
   seed.sql ข้อมูลจำลอง 20 ลีด ชื่อไทยสมมติ กระจายสถานะและช่องทาง
   → บอกให้ฉันไปวางใน Supabase SQL Editor แล้วเปิด Table Editor ดูว่ามีตารางและข้อมูลครบ
2. Backend (หลังบ้าน) — แก้เฉพาะ functions/api/
   join.js รับข้อมูลฟอร์ม ตรวจความถูกต้อง (ชื่อ เบอร์ ต้องมี · ต้องยินยอมให้ติดต่อ) แล้วสร้างลีดสถานะ new ด้วย
service_role key จาก env
   → ทดสอบส่งข้อมูลตัวอย่างเข้า /api/join แล้วรายงานผล
3. Frontend (หน้าบ้าน) — แก้เฉพาะ public/
   หน้า /join ฟอร์มสนใจสมัคร (ชื่อ เบอร์ อีเมล ประเภทที่สนใจ ช่องทาง ช่องยินยอมให้ติดต่อ [จำเป็น] แยกกับช่องยินยอมการ
ตลาด [ไม่บังคับ]) ส่งไป /api/join
   หน้า /app รายการลีด ค้นหา กรองตามสถานะ/ช่องทาง/ประเภท · หน้ารายละเอียดลีด เปลี่ยนสถานะ + ไทม์ไลน์ประวัติการติดต่อ
+ ช่องเพิ่มบันทึก
ใช้ชื่อตาราง คอลัมน์ และรูปแบบข้อมูลเดียวกันทั้ง 3 ชั้นตาม SPEC.md
```

2.3 สร้างตารางจริง (ทำเองบนเว็บ)

1. เปิด Supabase → **SQL Editor**
2. วาง `schema.sql` → กด **Run**
3. วาง `seed.sql` → กด **Run**
4. เปิด **Table Editor** เห็นตาราง = เห็น Database เป็นรูปธรรม

2.4 รันบนเครื่องและทดสอบ

PROMPT W2-RUN

Workshop 2.4 · รันบนเครื่องและทดสอบ

```
ฉันรัน schema.sql และ seed.sql ใน Supabase SQL Editor แล้ว
ช่วยรันเว็บบนเครื่องด้วย npx wrangler pages dev (ให้ Pages Functions ทำงานด้วย) แล้วบอกลิงก์ที่ต้องเปิด
จากนั้นบอกขั้นตอนทดสอบด่าน 1 ทีละข้อ:
1. กรอกฟอร์ม /join ด้วยข้อมูลทดสอบ → ลีดใหม่ต้องโผล่ในหน้า /app และใน Table Editor
2. เปลี่ยนสถานะลีดในหน้า CRM → ค่าใน Table Editor ต้องเปลี่ยนตาม
ถ้ามี error ให้อ่าน log แล้วแก้ให้ อธิบายสาเหตุสั้น ๆ
```

ด่าน 1 กรอกฟอร์ม `/join` ด้วยข้อมูลทดสอบ → ลีดโผล่ในหน้า CRM + ใน Table Editor · เปลี่ยนสถานะลีดในหน้า CRM → ค่าใน Table Editor เปลี่ยนตาม

2.5 ตั้งจุดย้อนกลับ

PROMPT W2-GIT

Workshop 2.5 · ตั้งจุดย้อนกลับด้วย Git

```
ผ่านด่าน 1 แล้ว ช่วยบันทึกจุดย้อนกลับด้วย Git ข้อความ "CRM v1: ฟอร์มสนใจสมัคร + หลังบ้านจัดการลีด"
ตรวจก่อนว่าไม่มีไฟล์ลับ (.dev.vars) ถูกบันทึกไปด้วย
```

WORKSHOP 3 · 15:15–16:30

W3 ระบบล็อกอินและบทบาท

เป้าหมาย: ล็อกอิน 3 บทบาท เห็นหน้าจอตาม `mockups/roles.html` และสิทธิ์ถูกล็อกทั้งที่หน้าจอและที่ฐานข้อมูล

3.1 สร้างบัญชีทดสอบ (ทำเองบนเว็บ)

1. Supabase → **Authentication → Users → Add user → Create new user**
2. เปิดตัวเลือก **Auto Confirm** ทุกครั้งที่สร้าง
3. สร้าง 3 บัญชี: `admin@club.test` · `sales@club.test` · `manager@club.test`

3.2 ผูกบัญชีกับบทบาท

ได้ SQL ไปวางใน SQL Editor

PROMPT W3-ROLES

Workshop 3.2 · ผูกบัญชีทดสอบกับบทบาท

```
ฉันสร้างบัญชีทดสอบใน Supabase Authentication แล้ว 3 บัญชี: admin@club.test · sales@club.test ·
manager@club.test
เขียน SQL ไฟล์ supabase/staff.sql ที่เพิ่มทั้ง 3 บัญชีลงตาราง staff พร้อมบทบาท admin / sales / manager (หา id
จาก auth.users ด้วยอีเมล ไม่ต้องให้ฉันคัดลอก id เอง)
และมอบหมายลีดจำลองครึ่งหนึ่งให้ sales@club.test
บอกฉันว่าต้องไปวางที่ SQL Editor แล้วกด Run
```

3.3 หน้าล็อกอิน + หน้าจอตามบทบาท

แนบภาพหน้าจอ หรืออ้างอิง `mockups/roles.html` (หน้าจอตัวอย่างใน Workshop 1.2)

PROMPT W3-LOGIN — Workshop 3.3 · หน้าล็อกอิน + หน้าจอตามบทบาท

ทำระบบล็อกอินด้วย Supabase Auth (อีเมล + รหัสผ่าน) และหน้าจอตามบทบาท ให้หน้าตาใกล้กับหน้าจอตัวอย่างที่แนบ **[แนบภาพหน้าจอจาก mockups/roles.html]**:
- /login หน้าเดียวสำหรับทุกบทบาท · ทุกหน้าใน /app ต้องล็อกอินก่อน ไม่งั้นเด้งกลับ /login
- หลังล็อกอิน อ่าน role จากตาราง staff แล้วเปิดเมนูตามบทบาท:
  แอดมิน — ลีดทั้งหมด (เพิ่ม/แก้/ลบ มอบหมายผู้ดูแล) · ทีมงานและสิทธิ์
  พนักงานขาย — ลีดของฉัน (เปลี่ยนสถานะ บันทึกการติดต่อ) · ไม่มีปุ่มลบ
  ผู้บริหาร — ภาพรวม (ตัวเลขสรุป + ลีดตามช่องทางและขั้นการขาย) · รายชื่อลีดแบบดูอย่างเดียว
- มุมล่างของเมนูแสดงชื่อและบทบาทของคนที่ล็อกอิน + ปุ่มออกจากระบบ
ทำทีละส่วน: ระบบล็อกอินก่อน แล้วหน้าจอทีละบทบาท หยุดให้ฉันทดสอบทุกส่วน

3.4 ล็อกสิทธิ์ที่ฐานข้อมูล (Row Level Security)



ซ่อนปุ่มไม่พอ ต้องล็อกตู้

หลักคิด

"ซ่อนปุ่มอย่างเดียวไม่พอ ต้องล็อกที่ตู้เก็บของด้วย"

PROMPT W3-RLS — Workshop 3.4 · ล็อกสิทธิ์ที่ฐานข้อมูล (Row Level Security)

ตอนนี้ซ่อนปุ่มที่หน้าจอแล้ว แต่ยังต้องล็อกที่ฐานข้อมูลด้วย
เขียน supabase/policies.sql ตั้ง Row Level Security ให้ตรงกับ docs/permissions.md:
- leads: แอดมินและผู้บริหารอ่านได้ทุกแถว · พนักงานขายอ่าน/แก้ได้เฉพาะแถวที่ assigned_to เป็นตัวเอง · ลบได้เฉพาะแอดมิน · ผู้บริหารแก้ไม่ได้
- contact_logs: เพิ่มได้เฉพาะแอดมินและพนักงานขาย (เฉพาะลีดของตัวเอง) · อ่านได้ตามสิทธิ์ของลีด
- staff: ทุกคนที่ล็อกอินอ่านได้ · แก้ได้เฉพาะแอดมิน
อธิบายทีละ policy เป็นภาษาคนสั้น ๆ แล้วบอกให้ฉันไปวางที่ SQL Editor

3.5 ทดสอบสิทธิ์

PROMPT W3-TEST

Workshop 3.5 · ทดสอบสิทธิ์ทั้ง 3 บทบาท

```
ทำเช็คลิสต์ทดสอบด่าน 2 จาก docs/permissions.md ให้ฉันกดทดสอบเองทีละบทบาท (admin / sales / manager)
และทดสอบที่ฐานข้อมูลโดยตรงด้วย: ล็อกอินเป็น sales แล้วลองสั่งลบลีดผ่าน supabase-js ใน console ของเบราว์เซอร์ — ต้องถูกปฏิเสธ
บอกโค้ด 1 บรรทัดที่ต้องวางใน console และผลที่ควรเห็น
```

ด่าน 2 สลับล็อกอินทั้ง 3 บทบาท แล้วเห็น/ทำได้ตรงกับ Permission Matrix — พนักงานขายลบไม่ได้แม้เรียกตรง · ผู้บริหารแก้ไม่ได้

WORKSHOP 4 · 16:30–17:00

W4 ขึ้นเว็บจริง + สรุป

4.1 Deploy

- ขึ้น Cloudflare Pages
- ตั้ง secret `SUPABASE_SERVICE_ROLE_KEY` ฝั่ง Cloudflare
- ใส่ URL เว็บใน Supabase → Authentication → URL Configuration

PROMPT W4-DEPLOY

Workshop 4.1 · ขึ้นเว็บจริงบน Cloudflare Pages

```
ช่วย Deploy โปรเจกต์นี้ขึ้น Cloudflare Pages:
- ชื่อโปรเจกต์ club-crm-[ชื่อเล่นของฉัน] · โฟลเดอร์หน้าเว็บ public/ · Functions อยู่ใน functions/
- ตั้งค่า SUPABASE_URL และ SUPABASE_SERVICE_ROLE_KEY เป็น secret ฝั่ง Cloudflare (ห้ามใส่ในโค้ด)
- บอกให้ฉันไปใส่ URL ของเว็บใน Supabase → Authentication → URL Configuration
- ให้ลิงก์เว็บจริง แล้วบอกขั้นตอนทดสอบด่าน 3 บนมือถือ
ถ้าต้องล็อกอิน Cloudflare ให้บอกฉันก่อน ฉันจะล็อกอินเอง
```

ด่าน 3 เปิด URL บนมือถือ → ล็อกอินได้ → กรอกฟอร์มได้

4.2 ปิดวัน

อัปเดตไฟล์บันทึกโปรเจกต์ว่าทำถึงไหน

PROMPT W4-WRAP

Workshop 4.2 · ปิดวัน อัปเดตไฟล์บันทึกโปรเจกต์

```
อัปเดตไฟล์บันทึกโปรเจกต์ (CLAUDE.md / AGENTS.md) ว่า:
- ตอนนี้ระบบทำอะไรได้แล้ว ผ่านด่านไหนแล้ว ลิงก์เว็บจริงคืออะไร
- ไฟล์สำคัญอยู่ที่ไหน
- งานต่อยอดที่ยังไม่ได้ทำ (จาก SPEC.md ข้อ 8)
แล้วบันทึกจุดย้อนกลับด้วย Git ข้อความ "Day 1 complete"
```

ทบทวนทั้งวัน

3 ชั้น
Frontend · Backend · Database

Use Case
บทบาท + Permission Matrix

Main Agent ทีละชั้น
Subagent ใช้ช่วยอ่าน ค้น ตรวจ

กติกา 3 ข้อ
ข้อมูลจำลอง · คีย์ลับไม่อยู่หน้าเว็บ · คนอนุมัติ

ก่อนนำไปใช้จริง (อ่านอย่างเดียว ไม่ทำในคลาส)
เปลี่ยนจากข้อมูลจำลองเป็นข้อมูลจริงต้องผ่านเช็คลิสต์ความปลอดภัย · นโยบาย PDPA · สำรองข้อมูล · รีเซ็ตรหัสผ่าน

NEXT STEPS

แผนต่อยอด

เพื่อให้ผู้เรียนเข้าใจพื้นฐานอย่างแท้จริงภายใน 1 วัน คลาสนี้สอนเฉพาะแกนกลางของระบบ ส่วนที่เหลือเป็นงานต่อยอดที่ผู้เรียนทำต่อเองได้ด้วยวิธีเดียวกัน



แกน CRM + โมดูลต่อยอด

#	งานต่อยอด	ต่อจากของวันนี้อย่างไร	PROMPT
1	**Join Now ขั้น 1–2: หน้า Explore + Ask AI + แบบประเมินความเหมาะสม %**	เพิ่มหน้าก่อนฟอร์ม `/join` ผลประเมินเก็บลงลีด	X1
2	**ขั้นอนุมัติแบบเต็ม + Audit Log**	เพิ่มสถานะ อนุมัติ/ปฏิเสธ/รอคิว + ตาราง `audit_logs`	X2
3	**Waiting List + จับคู่ซื้อ/ขาย/โอนสิทธิ์**	ตาราง `waitlist` / `listings` เชื่อมกับลีดและสมาชิก	X3
4	**ระบบจองห้อง/บริการ**	ตาราง `bookings` เชื่อมกับสมาชิก	X4
5	**ร่างสัญญาจากข้อมูลสมาชิก**	ดึงข้อมูลจาก leads/members ไปเติมแม่แบบ + ขั้นอนุมัติ	X5
6	**Chatbot / FAQ / LINE OA**	ตาราง `faqs` + แชตส่งลีดเข้า CRM เดิม	X6
7	**Agent ที่ทำงานในระบบ + แจ้งเตือนทีมขาย**	trigger ลีดใหม่ → AI สรุป/ร่างข้อความ → คนกดอนุมัติ	X7
8	**Dashboard ผู้บริหารแบบละเอียด + ล็อกอินขั้นสูง**	ต่อจากหน้าผู้บริหาร · ลืมรหัสผ่าน / Google / 2FA	X8

✕ นอกขอบเขตหลักสูตร

งานผลิตสื่อ · การตั้งเวลาโพสต์ · ค้นหา/เข้ากลุ่มออนไลน์อัตโนมัติ · ระบบชำระเงินออนไลน์ · ลายเซ็นอิเล็กทรอนิกส์

NEXT STEPS · PROMPT ตั้งต้น

ทุก Prompt เริ่มจาก "อ่าน SPEC.md" แล้วต่อยอดจากระบบที่ทำในวันนี้

PROMPT X1 — ต่อยอด 1 · Join Now ขั้น 1–2 + แบบประเมินความเหมาะสม

```
อ่าน SPEC.md แล้วเพิ่มขั้นก่อนฟอร์ม /join:
- หน้า Explore: รายละเอียดสโมสร ประเภทสมาชิก และสิทธิ์ ดึงจากฐานข้อมูล
- แบบประเมินความเหมาะสม: คำถามใช่/ไม่ใช่ คิดเป็น % ตามน้ำหนักที่ฉันกำหนด แสดงว่า "เป็นผลประเมินเบื้องต้น ไม่ใช่ผลอนุมัติ" ห้ามถามเรื่องโรคหรืออาการ
- เก็บผล % ลงลีดที่สร้าง
ถามน้ำหนักคะแนนจากฉันก่อน แล้วอัปเดต SPEC.md
```

PROMPT X2 — ต่อยอด 2 · ขั้นอนุมัติ + Audit Log

```
อ่าน SPEC.md แล้วเพิ่ม:
- ขั้นคนอนุมัติ: แอดมินกด อนุมัติ / ปฏิเสธ / เข้าคิวรอ พร้อมบันทึกเหตุผล ระบบไม่ตัดสินแทน
- ตาราง audit_logs บันทึกทุกการแก้ลีด: ใคร · เมื่อไหร่ · ค่าก่อน · ค่าหลัง (ใช้ trigger ในฐานข้อมูล) · แอดมินและผู้บริหารเปิดอ่านได้ แก้ไม่ได้
อัปเดต SPEC.md และ docs/permissions.md
```

PROMPT X3 — ต่อยอด 3 · Waiting List + จับคู่สิทธิ์

```
อ่าน SPEC.md แล้วออกแบบโมดูล Waiting List: ผู้รอสมัคร · ผู้ต้องการซื้อ/ขาย/โอน/เช่าช่วงสิทธิ์ · จับคู่ตามเงื่อนไข · แจ้งทีมเมื่อมีสิทธิ์ตรง · ประวัติการดำเนินการ
เสนอตารางใหม่และความเชื่อมกับ leads ให้ดูก่อน ยังไม่ต้องสร้าง
```

PROMPT X4

ต่อยอด 4 · ระบบจองห้อง/บริการ

อ่าน SPEC.md แล้วออกแบบระบบจอง: ตรวจห้อง/บริการว่าง · ตรวจสิทธิ์สมาชิก · ยืนยัน/เปลี่ยน/ยกเลิก · ประวัติการจองเชื่อมกับลีดและสมาชิก
เสนอตาราง bookings และหน้าจอให้ดูก่อน

PROMPT X5

ต่อยอด 5 · ร่างสัญญาจากข้อมูลสมาชิก

อ่าน SPEC.md แล้วทำโมดูลร่างสัญญา: เลือกสมาชิก → เติมข้อมูลลงแม่แบบสัญญา → แอดมินตรวจและกดอนุมัติก่อนพิมพ์ · เก็บประวัติเวอร์ชัน
ให้ AI ร่างได้ แต่คนต้องอนุมัติทุกฉบับ

PROMPT X6

ต่อยอด 6 · Chatbot + FAQ

อ่าน SPEC.md แล้วเพิ่มตาราง faqs (คำถาม · คำตอบ · ภาษา · สถานะตรวจแล้ว) และแชตบอตบนหน้าเว็บที่ตอบจาก FAQ ที่ตรวจแล้วเท่านั้น
ไม่รู้คำตอบให้ส่งต่อเจ้าหน้าที่ และถ้าลูกค้าสนใจสมัครให้สร้างลีดเข้า CRM เดิม

PROMPT X7

ต่อยอด 7 · Agent ที่ทำงานในระบบ

อ่าน SPEC.md แล้วทำ Agent ที่ทำงานในระบบ: เมื่อมีลีดใหม่ (trigger) → AI สรุปความสนใจและร่างข้อความติดต่อ → บันทึกลงฐานข้อมูล → แจ้งพนักงานขายที่ได้รับมอบหมาย → พนักงานอ่าน แก้ แล้วกดส่งเอง
คีย์ AI ต้องอยู่ฝั่ง Backend เท่านั้น · AI ห้ามส่งข้อความหาลูกค้าเอง

PROMPT X8

ต่อยอด 8 · Dashboard ละเอียด + ล็อกอินขั้นสูง

อ่าน SPEC.md แล้วเพิ่ม:
- Dashboard ผู้บริหาร: ลีดรายสัปดาห์ · อัตราสมัครตามช่องทาง · เวลาตอบลีดเฉลี่ย · ลีดค้างเกิน 24 ชม.
- ล็อกอินขั้นสูง: ลืมรหัสผ่าน · ล็อกอินด้วย Google
ถามก่อนว่าต้องการตัวเลขไหนบ้าง

TROUBLESHOOTING

แก้ปัญหาที่พบบ่อย

อาการ	สาเหตุที่พบบ่อย	ทางแก้
ฟอร์มกดส่งแล้วขึ้น error / ลีดไม่เข้า	ยังไม่ได้ Run `schema.sql` หรือคีย์ใน `.dev.vars` ผิด	เปิด Table Editor ดูว่ามีตาราง leads · ให้ Agent ตรวจ `.dev.vars` (ห้ามส่งคีย์ลับในแชตสาธารณะ)
ล็อกอินแล้วเห็นหน้าว่าง / ไม่มีลีด	RLS ล็อกถูกต้อง แต่บัญชียังไม่ได้ผูกบทบาทในตาราง staff	รัน SQL จาก PROMPT W3-ROLES อีกครั้ง
พนักงานขายไม่เห็นลีดเลย	ลีดยังไม่ได้มอบหมาย (`assigned_to` ว่าง)	ล็อกอินแอดมิน มอบหมายลีดก่อน
ล็อกอินบนเว็บจริงแล้วเด้งกลับ	ยังไม่ได้ใส่ URL เว็บใน Supabase URL Configuration	ใส่ URL `*.pages.dev` แล้วลองใหม่
Agent แก้ไฟล์มั่ว / ทำเกินที่สั่ง	SPEC หรือไฟล์กติกาไม่ชัด	สั่งย้อนกลับจุด Git ล่าสุด แล้วเพิ่มกติกาให้ชัด
Agent แก้หลายชั้นพร้อมกันจนงง	สั่งงานกว้างเกินไป	สั่งให้ทำทีละชั้น และหยุดให้ทดสอบก่อนไปชั้นถัดไป
เผลอวางคีย์ลับในไฟล์หน้าเว็บ	—	ลบออก แล้ว Rotate key ใหม่ใน Supabase ทันที

QUICK REFERENCE

Prompt Cheat Sheet

รายชื่อ Prompt ทั้งหมดในคู่มือ เรียงตามลำดับการใช้ — เปิดหน้าของขั้นนั้นแล้วกดคัดลอก

PROMPT	ขั้น	ใช้ทำอะไร
S1	Setup 1	เช็คเครื่องมือในเครื่อง
S3	Setup 3	ตั้งโฟลเดอร์โปรเจกต์ + Git
S2-EXPLAIN	Session 2	ให้ Agent อธิบายระบบเป็น 3 ชั้น
W1-USECASE	Workshop 1.1	วาด Use Case Diagram
W1-MATRIX	Workshop 1.3	ทำตารางสิทธิ์ (Permission Matrix)
W1-DATA	Workshop 1.4	ออกแบบข้อมูล 3 ตาราง (ยังไม่สร้าง)
W1-SPEC	Workshop 1.5	เขียน SPEC.md
W1-RULES	Workshop 1.6	เขียนไฟล์กติกาทีม (CLAUDE.md / AGENTS.md)
W2-CONNECT	Workshop 2.1	เชื่อมโปรเจกต์กับ Supabase
W2-BUILD	Workshop 2.2	Main Agent สร้างทีละชั้น (Database → Backend → Frontend)
W2-RUN	Workshop 2.4	รันบนเครื่องและทดสอบ
W2-GIT	Workshop 2.5	ตั้งจุดย้อนกลับด้วย Git
W3-ROLES	Workshop 3.2	ผูกบัญชีทดสอบกับบทบาท
W3-LOGIN	Workshop 3.3	หน้าล็อกอิน + หน้าจอตามบทบาท
W3-RLS	Workshop 3.4	ล็อกสิทธิ์ที่ฐานข้อมูล (Row Level Security)
W3-TEST	Workshop 3.5	ทดสอบสิทธิ์ทั้ง 3 บทบาท
W4-DEPLOY	Workshop 4.1	ขึ้นเว็บจริงบน Cloudflare Pages
W4-WRAP	Workshop 4.2	ปิดวัน อัปเดตไฟล์บันทึกโปรเจกต์
X1	ต่อยอด 1	Join Now ขั้น 1–2 + แบบประเมินความเหมาะสม
X2	ต่อยอด 2	ขั้นอนุมัติ + Audit Log
X3	ต่อยอด 3	Waiting List + จับคู่สิทธิ์
X4	ต่อยอด 4	ระบบจองห้อง/บริการ
X5	ต่อยอด 5	ร่างสัญญาจากข้อมูลสมาชิก
X6	ต่อยอด 6	Chatbot + FAQ
X7	ต่อยอด 7	Agent ที่ทำงานในระบบ
X8	ต่อยอด 8	Dashboard ละเอียด + ล็อกอินขั้นสูง

ติดต่อสถาบัน ZenityX

โทร
062-262-9644

อีเมล
admin@zenityxai.com

ออกแบบหลักสูตรโดยสถาบัน ZenityX · First Vibe Coding — สร้างระบบ CRM สโมสรแบบพื้นฐาน · AI Agent Co-Working System · Day 1 · ข้อมูลทั้งหมดในคู่มือเป็นข้อมูลจำลองเพื่อการเรียนการสอน Harmony Club เป็นชื่อสมมติ