

LINE Rich Menu Web

ระบบเว็บไซต์สำหรับให้ผู้ใช้ล็อกอินด้วย LINE และสามารถปรับแต่งและใช้งาน Rich Menu ที่ต้องการใช้งานได้เอง

Link to line-rich-menu-web : <https://line-rich-menu-web.onrender.com>

Link to Github : <https://github.com/antcpozxc-netizen/Line-Rich-Menu-Web.git>

Link to Figma :

<https://www.figma.com/design/WvACoGf7Hr7Nvd75kLJTjf/Line-Rich-Manus-Web?node-id=0-1&t=4J9Rwxv5BKkubUE8-1>

วัตถุประสงค์ของโครงการ

- เพื่อพัฒนา Web Application ที่สามารถให้ผู้ใช้ LINE Official Account เปลี่ยนและปรับแต่ง Rich Menu ด้วยตนเอง
- เพื่อให้ผู้ใช้งานสามารถเลือกเปลี่ยนเมนูหรือสิ่งงานเมนูผ่าน Web Application
- พัฒนา Web Application ที่เชื่อมต่อกับ LINE Messaging API เพื่อจัดการ Rich Menu
- ให้ผู้ใช้ที่มีบัญชี LINE Official Account สามารถสร้างและเปลี่ยน Rich Menu ได้เองผ่านระบบ
- เพิ่มความสะดวกและประสิทธิภาพในการจัดการเมนูสำหรับเจ้าของ LINE OA
- เพื่อศึกษาการทำงานของ LINE Messaging API และ LINE Login API
- สร้าง Bot จัดการการสั่งงานติดตามงาน

Requirement ของระบบ (Functional)

ระบบ Web Application สำหรับเปลี่ยน Rich Menu ของ LINE มีฟังก์ชันหลัก ดังนี้

การเข้าสู่ระบบด้วยบัญชี LINE

- ผู้ใช้เข้าสู่ระบบผ่าน LINE Login API
- ระบบดึงและตรวจสอบ `userId` จาก LINE หลังการ Login
- ตรวจสอบความถูกต้องของ token ก่อนอนุญาตใช้งานฟีเจอร์

การจัดการ Rich Menu

- สร้าง Rich Menu ใหม่ กำหนดชื่อเมนู ขนาด และปุ่มเมนูพร้อม action
- ออกแบบ Design image เพื่อนำไปใช้งานที่ rich menu ได้
- อัปโหลดภาพพื้นหลังเมนูตามขนาดที่ LINE กำหนด
- แสดงรายการ Rich Menu ทั้งหมดของบัญชี OA
- เลือกและตั้งค่าการใช้งาน Rich Menu ผ่าน API
- เมนูย่อยในการเลือกการทำงาน(Action/ event) ของปุ่ม
 - Q&A → เชื่อมกับระบบ FAQ หรือข้อความตอบอัตโนมัติ

- Live Chat → เปิดแชทกับแอดมิน (ส่งข้อความไปยัง webhook หรือแจ้งเตือนแอดมิน)
- เปิดเว็บไซต์
- ส่งข้อความข้อความเฉพาะ
- เปิด Flex Message

การส่ง Broadcast Message

- เขียนข้อความ / แบบรูป
- เลือกกลุ่มเป้าหมายหรือทั้งหมด
- ส่งข้อความผ่าน API ได้

การจัดการสิทธิ์และความปลอดภัย

- จำกัดการเข้าถึงฟีเจอร์ให้เฉพาะผู้ที่เชื่อมบัญชี OA และมีสิทธิ์
- เก็บรักษา Channel Access Token อย่างปลอดภัยในระบบ Backend
- ใช้ระบบ Authentication และ Authorization ภายใน Web App

การบันทึกและดูรายงาน/สถิติการใช้งาน แสดงประวัติการเปลี่ยนแปลง

- บันทึกข้อมูลการสร้างและเปลี่ยนแปลง Rich Menu
- แสดงประวัติการใช้งานให้ผู้ใช้ตรวจสอบได้
- ดึงข้อมูลการคลิก Rich Menu
- ดูอัตราการเปิดข้อความ Broadcast
- แสดงผ่านกราฟ/ตาราง

Note : อาจจะลองหาวิธีการจัดการเรื่องความปลอดภัยของ Channel Access Token เพิ่มเติมครับ

เทคโนโลยีที่ใช้ (Technology Stack / Tools)

ประเภท	รายละเอียด
Frontend	HTML, JavaScript (React.js)
Backend	Node.js (Express.js)
API	LINE Messaging API, LINE Login API
Database	Firebase
Hosting	Firebase Hosting, Render หรือเครื่องเซิร์ฟเวอร์ส่วนตัว
อื่น ๆ	Postman (สำหรับทดสอบ API), Ngrok (ทดสอบ Webhook local)

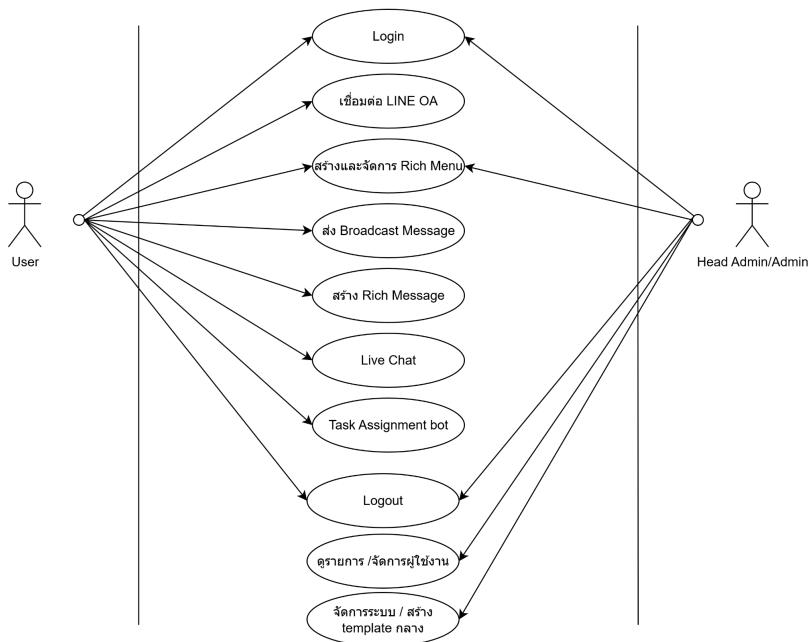
ข้อมูลหรือ Tools ที่ใช้อาจมีการเปลี่ยนแปลงภายหลัง

ตารางพัฒนาโครงการ LINE Rich Menu Switcher (2 สัปดาห์ หรืออาจ 3 สัปดาห์ครับ)

วันที่	งานที่ทำ	รายละเอียด
Day 1-2	วิเคราะห์ระบบ + วางแผน	<ul style="list-style-type: none">- วิเคราะห์ความต้องการ- วิเคราะห์ Use Case, Flow Diagram- วางแผน Project + Tool ที่ใช้
Day 3	ออกแบบ Database	<ul style="list-style-type: none">- วิเคราะห์และออกแบบฐานข้อมูล โดยใช้ Firebase
Day 4	ออกแบบ UI + เตรียม LINE Developer	<ul style="list-style-type: none">- สเก็ต Mockup / UI หน้า Web- สร้าง LINE Channel บน LINE Developers Console- จดบันทึก Channel ID, Secret, Access Token
Day 5-6	เริ่มพัฒนา Frontend (HTML + JS)	<ul style="list-style-type: none">- หน้า Login (ปุ่ม LINE Login)- หน้าแสดงรายการ Rich Menu
Day 6-7	เริ่มพัฒนา Backend	<ul style="list-style-type: none">- สร้าง Server (Node.js/Express)- เตรียม Route สำหรับเชื่อม API กับ LINE
Day 8	เชื่อม LINE Login API เข้ากับ Web	<ul style="list-style-type: none">- ผู้ใช้สามารถ Login ได้- ได้ userId ของ LINE มาที่ backend
Day 9	ดึง Rich Menu จาก LINE	<ul style="list-style-type: none">- GET /v2/bot/richmenu/list- แสดงเมนูทั้งหมดใน Web UI
Day 10	สร้างฟังก์ชันเปลี่ยน Rich Menu	<ul style="list-style-type: none">- POST /v2/bot/user/{userId}/richmenu/{richMenuId}- ผู้ใช้กดเลือกเมนู และระบบส่งคำสั่งให้ LINE
Day 11-12	สร้าง template Rich Menu และปรับแต่งฟังก์ชันอื่นๆ	<ul style="list-style-type: none">- สร้าง template Rich Menu สำหรับ User และ Admin (Template กลาง)

Day 13-14	เปลี่ยนรูปแบบการ Login และ การใช้งานก่อน Login + ทดสอบระบบ	- เพิ่ม flow การ login การเข้าใช้งานก่อน login และเพิ่ม Line OA
Day 15	ทดสอบการทำงานของระบบ + เพิ่มคู่มือการใช้งาน	- ทดสอบระบบสร้าง Rich menu และนำไปใช้งาน
Day 16	เพิ่มระบบ Manage Admin เพื่อจัดการการทำงานของ user	- เพิ่มระบบ Manage Admin เพื่อจัดการการทำงานของ user
Day 17	ทดสอบระบบการใช้งาน + เริ่ม ระบบปรับแต่ง Rich Menu เพิ่มเติม	- ทดสอบระบบ Admin - สร้างระบบ Design image Rich Menu
Day 18	ทดสอบระบบการใช้งานระบบ ปรับแต่ง Rich Menu เพิ่มเติม	- ทดสอบระบบ Design image Rich Menu
Day 19	จัดการข้อมูล สร้างเอกสารคู่มือ	- ตรวจสอบข้อมูล เอกสารคู่มือ คู่มือใช้งาน, คู่มือติดตั้ง

Use Case Diagram



1. User

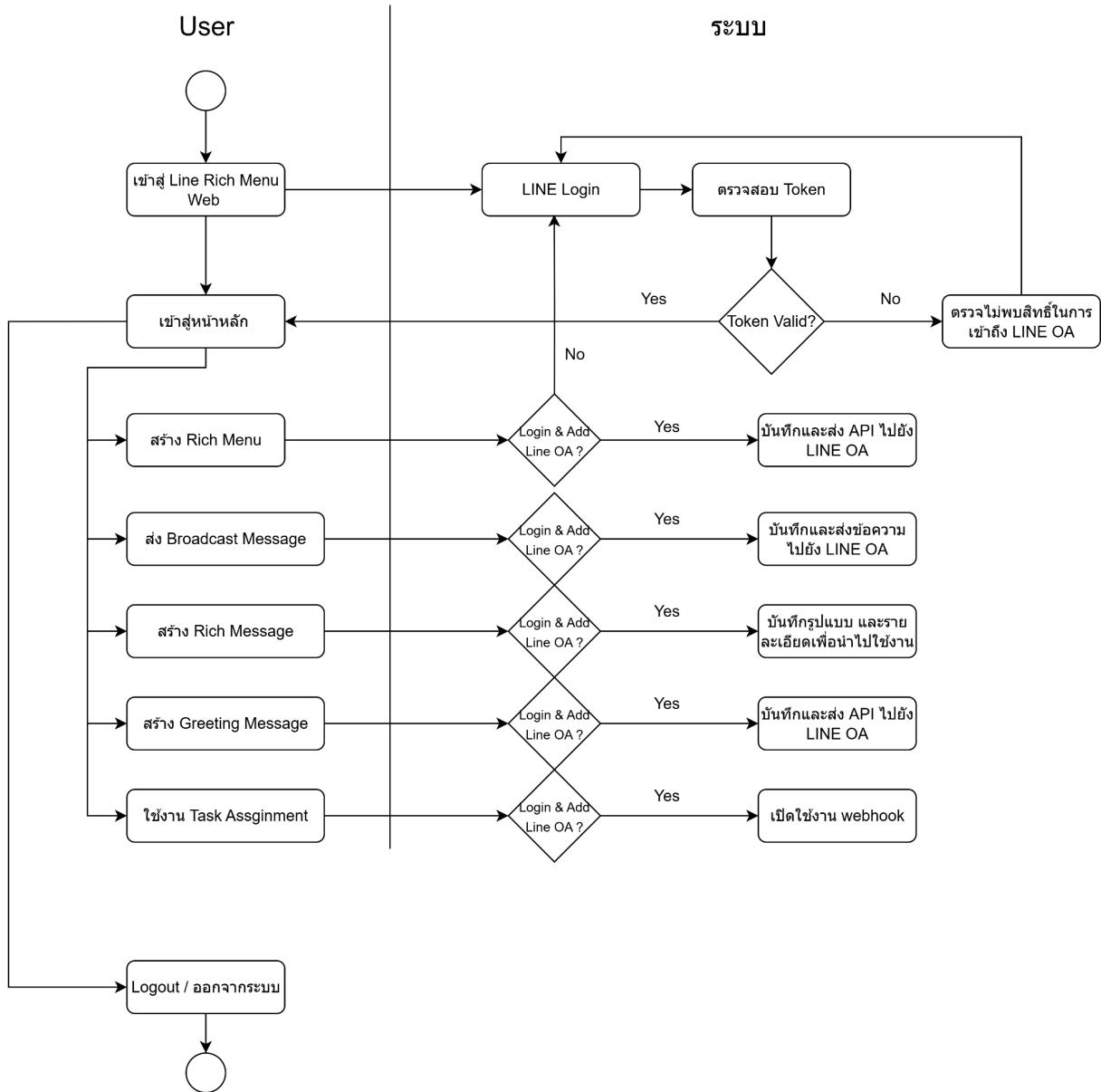
Use Case	คำอธิบาย
Login	เข้าสู่ระบบผ่าน LINE Login หรือระบบบัญชีในเว็บ
เชื่อมต่อ LINE OA	เชื่อมต่อนับบัญชี LINE OA ผ่าน Channel Access Token
สร้าง Rich Menu	สร้างและจัดการ Rich Menu (อัปโหลดรูป, กำหนด action, preview) และเมนูย่อย
ส่ง Broadcast Message	ส่งข้อความแบบ Broadcast ไปยัง Follower ทุกคน
สร้าง Rich Message	สร้าง Rich Message เพื่อนำไปใช้งานที่ Broadcast

ตั้งค่า Greeting Message	Set รูปแบบข้อความเมื่อผู้ใช้งานเพิ่มเพื่อนใน Line ครั้งแรก
Live Chat	เปิด session พูดคุยกับ user ที่เข้า LINE OA เมื่อผู้ใช้งานกด action ที่ Rich Menu (action = Live chat)
Task Assignment bot	ใช้งาน bot ติดตามงาน กับ line oa
Logout	ออกจากระบบ

2. Head Admin / Admin

Use Case	คำอธิบาย
ดู/จัดการรายการผู้ใช้งาน	ดูรายชื่อผู้ใช้ที่เข้า LINE OA เข้ากับระบบ
จัดการระบบ Template Rich Menu	สร้างและตั้งค่า default Template Rich Menu

Flow Diagram



Flow Diagram การทำงานแต่ละส่วน

1. การเชื่อมต่อ LINE OA (ครั้งแรก) เพื่อเข้าใช้งาน

[User] : - เปิดเว็บไซต์

- Login ผ่าน LINE OAuth หรือระบบในเว็บ
- กด "เชื่อมต่อ LINE OA"
- รอก Channel Access Token (Long-lived)
- [ระบบ] ตรวจสอบ Token validity ผ่าน LINE API
- [ระบบ] บันทึกข้อมูลผู้ใช้และ Token ไว้ในฐานข้อมูล
- เข้าสู่หน้าหลัก

2. การสร้าง Rich Menu

[User] - เลือกเมนู "สร้าง Rich Menu"

- (หากไม่มีรูป) สามารถเลือก template ที่ต้องการและกด Design image ได้
- ตั้งชื่อ Rich Menu
- เลือกขนาด (เช่น 2500x843)
- เพิ่มปุ่ม (action เช่น ส่งข้อความ, เปิดลิงก์)
- อัปโหลดรูป
- Preview Rich Menu
- กด "เผยแพร่"
- [ระบบ] เรียก LINE API สร้าง Rich Menu และเชื่อมกับ User

3. การส่ง Broadcast Message

[User] - เลือกเมนู "Broadcast"

- พิมพ์ข้อความหรือแนบรูป
- กด "ส่งข้อความ"
- [ระบบ] เรียก LINE API (Push หรือ Multicast)
- [ระบบ] เก็บ log การส่งข้อความ

4. การสร้าง Rich Message

[User] - เลือกเมนู "Rich Message"

- แนบรูปและตั้งค่าพื้นที่ action ว่ากดแล้วจะมี action แบบไหน
- กด Publish to Image Map
- [ระบบ] เก็บข้อมูลการสร้าง Rich Message

5. การตั้งค่า Greeting Message

[User] - เลือกเมนู "Greeting Message"

- เพิ่มหรือแก้รายละเอียดหัวข้อความรูปภาพและอื่นๆ ที่เมื่อเพิ่มผู้ใช้งานครั้งแรกจะส่งไปให้
- [ระบบ] เก็บข้อมูลและส่งเมื่อมีการเพิ่มผู้ใช้งานครั้งแรก

6. Live Chat

[User] - เมื่อ set action ที่ Rich Menu ให้เป็น Live Chat และเมื่อผู้ใช้งานกด action Live chat จាតทำการ เปิด session chat โดยผู้ที่มี Line OA เดียวกับที่ user กด Live chat จะสามารถพิมพ์คุยผ่าน menu “Live Chat” ได้โดยตรง

7. Add Template Rich Menus [Developer/ Head Admin/ Admin]

- เลือกเมนู "Add Template Rich Menus"
- สามารถสร้าง Template Rich Menus เพื่อให้ผู้ใช้งานทุกคนเลือกนำมาใช้งานได้

8. Rich Menu Designer

- รองรับ Layers ต่อบล็อก (ประเภท text/image/sticker) → สร้างหลายชั้นในบล็อกเดียวได้, ซ่อน/แสดง, คัดลอก, ล่าดับชั้น (นำชั้น/ลง)
- Highlight บล็อกที่กำลังแก้ + เลือกจาก Preview ได้
- Auto-fit รูป ให้เต็มบล็อก (cover) + ปรับครอบ/ตำแหน่ง
- เลือกฟอนต์ / เงา สำหรับตัวอักษร
- Sticker library (SVG data-URL + ไฟล์ใน src/assets/stickers) พร้อมชื่อสติกเกอร์
- Undo/Redo: มีอยู่ แต่อาจยังไม่ครอบคลุมทุก action (แนะนำบันทึกเป็น “known limitation”)
- Export: ออกภาพตามสเปก LINE ที่เลือก (เช่น 2500×843/1686) ให้ระบุชัดเจนว่า “ภาพ Export = มุ่งมองจริง”

9. Task Assignment

[User] - เข้าเมนู Task Assignment

1. กรอก Google Sheet ID
2. กด save และกด Verify connection (ตรวจสอบการเชื่อมต่อ)
3. กด enable เพื่อใช้งาน

10. Administrator management [Developer/ Head Admin/ Admin]

ที่ Home → เข้าใช้งานด้วย Role Admin (ขั้นต่ำ) → Administrator management

ตารางถูกแบ่งเป็น 3 ส่วน:

1. Developers
2. Head admins & Admins
3. Users

การเปลี่ยนบทบาท (Role)

1. หาแคล้วใช้ที่ต้องการ
2. เลือกจาก Dropdown (user / admin / head Admin / developer)
3. หรือกดปุ่ม SET USER เพื่อรีเซ็ตเป็น user ทันที
4. ระบบจะบันทึกทันทีและรีโหลดรายการ (มีแจ้งสถานะ error ด้านบนหากทำไม่สำเร็จ)

บันทึกสำคัญ:

- Developer ไม่สามารถลดชั้นตัวเอง (ปุ่ม/เมนูจะถูกปิดไว้)
- การเปลี่ยนบทบาทบางครั้งอาจต้อง รีเฟรชหน้า หรือ Sign out / Sign in ใหม่ เพื่อให้สิทธิ์หน้าอื่นๆ อัปเดตครบ

การลบผู้ใช้ (Delete)

- Developer: ตั้งบทบาททุกคน/ลบได้ทุกคน (ยกเว้นลบตัวเอง)
- Head Admin: ลบ user 'ได้, ตั้ง/ลดชั้น admin & user 'ได้ (ลบ admin/หัวหน้า/นักพัฒนาไม่ได้)
- Admin: ลบ user 'ได้, เพิ่ม admin 'ได้ (ลบ admin/หัวหน้า/นักพัฒนาไม่ได้)

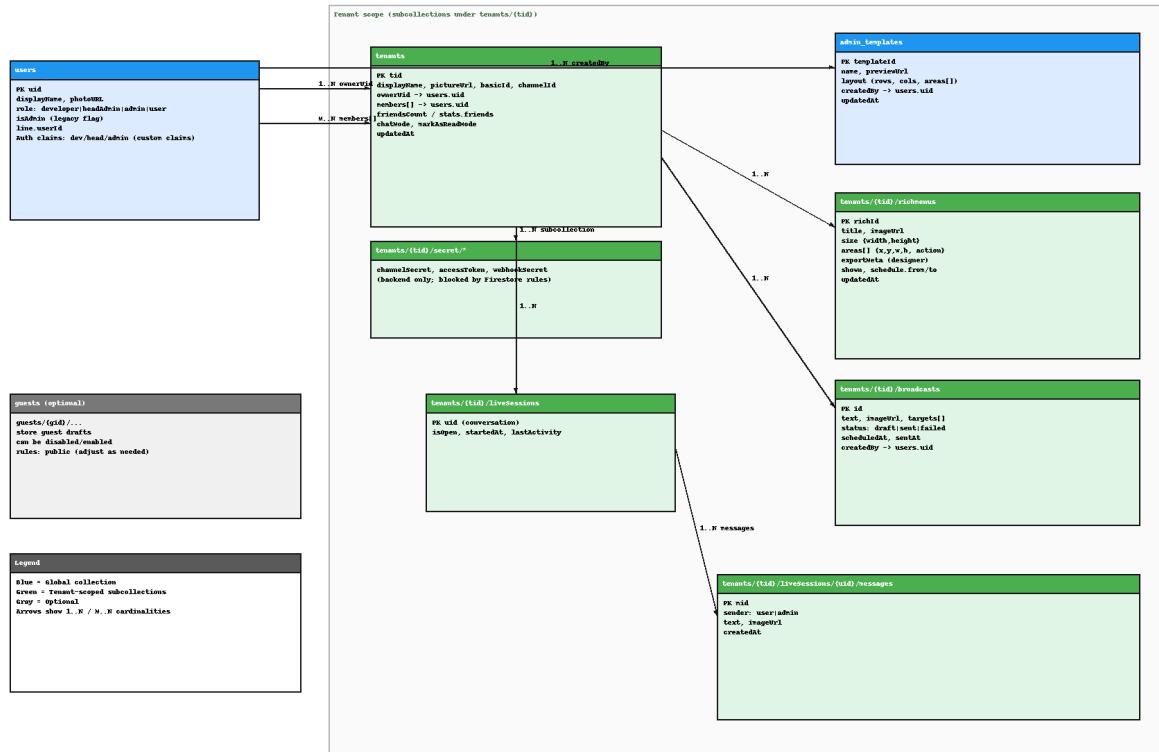
ข้อแนะนำในการใช้งาน

- ใช้ช่อง Role (Dropdown) สำหรับ เลื่อนขั้น/ลดขั้น อย่างเป็นทางการ
- ใช้ปุ่ม SET USER สำหรับรีเซ็ตค่าที่ไม่ต้องมีสิทธิ์แอดมินแล้ว
- แนะนำให้มี Developer อย่างน้อย 1 คนเสนอ ในการต้องกู้สิทธิ์
- ถ้าเปลี่ยนบทบาทแล้วเมนูซ้าย (Admin / Template ฯลฯ) ยังไม่เปลี่ยน ให้รีเฟรชหน้า หรือ ออกเข้าใหม่

ตารางสรุปสิทธิ์

ความสามารถ	Developer	Head Admin	Admin	User
เห็นเมนู Admin	✓	✓	✓	-
ตั้ง/ลดสิทธิ์ทุกบทบาท	✓	-	-	-
ตั้ง/แก้สิทธิ์ Admin & User	✓	✓	✓ (เพิ่ม admin ได้)	-
ลบ User	✓	✓	✓	-
ลบ Admin/Head/Dev	✓ (ยกเว้นลบตัวเอง)	-	-	-

ER Diagram



โครงสร้างข้อมูล Firestore

users/{uid} ใช้ uid จาก Firebase (สำหรับ LINE จะเห็นเป็นรูปแบบ line:Uxxxxxxxxx)
tenants/{tid} คือ OA แต่ละตัว (ผูกกับ owner + members)

1) users/{uid}

ฟิลด์	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
displayName	string	≤ 120	"Po"
photoURL	string(URL)	≤ 2,083	"https://..."
line.userId	string	≤ 64	"Uabc..."
isAdmin	boolean	-	true/false
role	string enum	developer/headAdmin/admin/user	"admin"
updatedAt	timestamp	-	server time

Custom claims (Auth): dev, head, admin (ใช้ตรวจสอบผู้ใช้งานเวอร์/แอป)

2) tenants/{tid}

ฟิลด์	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
displayName	string	≤ 120	"@Borcelle Cafe"
pictureUrl	string(URL)	≤ 2,083	"https://..."
basicId	string	≤ 100	"@123abcd"
channelId	string	≤ 64	"2007922762"
ownerUid	string	≤ 128	"line:Uxxxx"
members	array<string>	รายการ uid ≤ 500	["line:U1", "line:U2"]
friendsCount/stats.friends	number	≥ 0	1234
chatMode / markAsReadMode	string/bool	optional	-
updatedAt	timestamp	-	server time

2.1 tenants/{tid}/secret/* (ห้าม client อ่าน/เขียนด้วย rules)

ฟิลด์	ชนิด	หมายเหตุ
channelSecret	string	เก็บเฉพาะฝั่ง backend
accessToken	string	Long-lived token OA
webhookSecret	string	ถ้ามี

Rules ปิดอ่าน/เขียน subcollection secret จาก client

3) admin_templates/{templateId}

ฟิลด์	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
name	string	≤ 120	"Cafe – 2x2 + header"
previewUrl	string(URL)	≤ 2,083	"https://..."
layout	map	JSON กำหนดกรอบ/ช่อง	{ rows, cols, areas[] }
createdBy	string	≤ 128	"uid"
updatedAt	timestamp	-	server time

อ่านได้ทุกคน (request เท่านั้น), เขียนได้เฉพาะ isAdmin==true

4) tenants/{tid}/richmenus/{richId}

ฟิลด์	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
title	string	≤ 120	"Main menu"
size	map	width, height	{ "width":2500, "height":843 }
areas	array<map>	≤ 20; แต่ละอัน {x,y,w,h, action}	ดูด้านล่าง
imageUrl	string(URL)	≤ 2,083	"https://..."
shown	boolean	-	true/false
schedule.from / schedule.to	timestamp	optional	-
exportMeta	map	ข้อมูลจาก Designer (layers, font ฯลฯ)	JSON
updatedAt	timestamp	-	server time

โครง areas[]

```
{ "x":0, "y":0, "w":1250, "h":843,
  "action": { "type":"uri|message|postback|richmessage",
    "data": { ... } } }
```

ขนาดภาพที่ LINE รองรับ: 2500x843 (หรือ 2500x1686)

ผู้ส่ง Storage เก็บที่: tenants/{tid}/rich-menus/<filename>.jpg|png

5) tenants/{tid}/broadcasts/{docId}

พิล็ต	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
text	string	≤ 5000	เนื้อความ
imageUrl	string(URL)	≤ 2,083	"https://..."
targets	array<string>	รายชื่อ userId (optional)	["Uxxx"]
status	string enum	"draft"	"sent"
scheduledAt	timestamp	optional	-
sentAt	timestamp	optional	-
createdBy	string	≤ 128	"uid"

6) Live chat

tenants/{tid}/liveSessions/{uid}

พิล็ต	ชนิด	ตัวอย่าง
isOpen	boolean	true
lastActivity	timestamp	now
startedAt	timestamp	now

tenants/{tid}/liveSessions/{uid}/messages/{mid}

พิล็ต	ชนิด	ขนาด/ข้อจำกัด	ตัวอย่าง
sender	string enum	"user"	"admin"
text	string	≤ 5000	"Hello"
imageUrl	string(URL)	≤ 2,083	optional
createdAt	timestamp	-	now

คู่มือการใช้งาน Line Rich Menus Web

สารบัญ

- ภาพรวมระบบ
- เริ่มต้นใช้งาน & การเข้าสู่ระบบ
- การเชื่อมต่อ LINE OA (Channel ID / Channel secret)
- หน้าจอ Accounts (เพิ่ม OA / จัดการสมาชิก)
- หน้าจอ Broadcast (ส่งข้อความครั้งล่มาก ๆ)
- หน้าจอ Rich Message (Imagemap/พื้นที่กด)
- หน้าจอ Greeting Message (ข้อความต้อนรับ)
- หน้าจอ Rich Menu (เมนูภาคลึกได้) + Action QnA
- หน้าจอ Template Rich Menus สำหรับแอดมิน
- หน้าจอ Task Assignment
- การกำหนดสิทธิ์ Admin ให้ผู้ใช้
- แนวทางแก้ปัญหา (Troubleshooting)
- ภาคผนวก (สเปคไฟล์, ตัวอย่าง Payload/Rule)

1) ภาพรวมระบบ

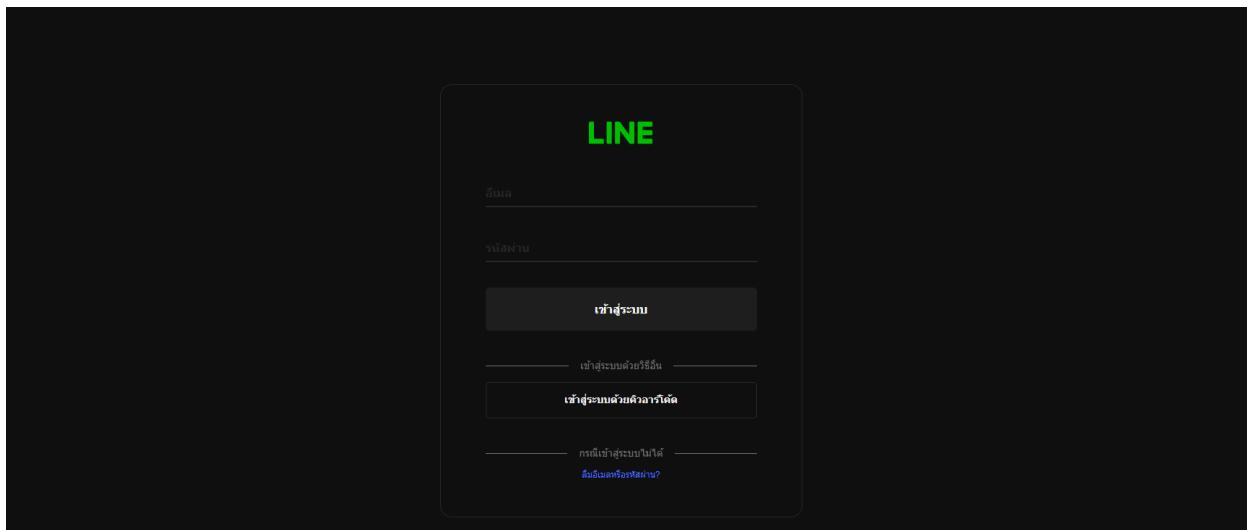
- แพลตฟอร์มช่วย ออกแบบ/ส่งคอนเทนต์ให้ LINE OA: Broadcast, Rich Message, Greeting และ Rich Menu
- รองรับ 2 โหมดการใช้งาน
 - Guest:** เก็บแบบร่างในเครื่องข้าวครัว (localStorage) — เมื่อกด "ส่ง/บันทึกข้อ OA" ระบบจะพาไป Login อัตโนมัติ
 - Logged-in:** ทำงานเต็มรูปแบบ, อัปโหลดไฟล์ขึ้น Firebase Storage และบันทึกข้อมูลจริง



2) เริ่มต้นใช้งาน & การเข้าสู่ระบบ

- ปุ่ม เข้าสู่ระบบด้วย LINE จะพาไปยัง LINE OAuth อย่างเป็นทางการ
- และปุ่มเยี่ยมชม (โหมด Guest) พาเข้าหน้า Home แต่จะยังไม่สามารถใช้งานฟังก์ชันต่างๆได้

หน้าแรกของ “LINE RICH MENUS WEB”



3) การเชื่อมต่อ LINE OA (Channel ID / Channel secret)

ใช้ข้อมูลจาก **Messaging API** ใน LINE Developers เพื่อให้ระบบผูกกับ OA ของคุณ

หมายเหตุที่ใน

- A) มี Line Official Account อญแล้ว

1. เข้าสู่ [LINE Official Account Manager](#) และเลือกเลือก Account ที่ต้องการจาก List

The screenshot shows the 'Accounts' section of the LINE Official Account Manager. On the left, there's a sidebar with 'Accounts' and 'Create new' options. The main area displays a table titled 'Accounts (5)' with columns for 'Account name', 'Friends', 'Role', and 'Plan'. The accounts listed are: 'test' (Friends: 1, Role: Administrator, Plan: ฟรี), 'Rich Menu Web' (Friends: 2, Role: Administrator, Plan: ฟรี), 'Test Web' (Friends: 1, Role: Administrator, Plan: ฟรี), 'Test Menu' (Friends: 1, Role: Administrator, Plan: ฟรี), and another 'test' account (Friends: 1, Role: Administrator, Plan: ฟรี). There are navigation arrows at the bottom of the table.

2. กดไปที่ setting หมุนบนขวา

The screenshot shows the 'Rich Menu Web' account page in the LINE Official Account Manager. The left sidebar includes options like Broadcast, Step messages, Auto-response messages, Rich media messages, Outreach, Chat screen, Data controls, Audiences, and Tracking (LINE Tag). The main content area has sections for Broadcasts, Content, To-do list (with a 'Making friends' goal of 1/5), Announcements, and a promotional banner for LINE OA. A red box highlights the 'Settings' button in the top right corner, which is also indicated by a red arrow pointing from the previous step's note.

3. ไปที่หัวข้อ Messaging API และกด Enable Messaging API

The screenshot shows the 'Settings' section of the LINE Official Account Manager. Under 'Messaging API', it says 'Status: Disabled'. A large green button labeled 'Enable Messaging API' is prominently displayed.

4. เลือก Provider ที่ต้องการ หรือสร้าง Provider ใหม่

A modal dialog box titled 'Select provider' is open. It asks 'Please select the company or person who operates this account.' Below this, it says 'A provider is an individual developer, company, or organization that manages users' personal information to offer various services.' There is a note about agreeing to the 'Messaging API Terms and Conditions of Service'. At the bottom, there are 'Cancel' and 'Agree' buttons.

5. กด OK

A modal dialog box titled 'Enabling Messaging API' is open. It contains a message: 'You won't be able to change or unlink this provider once linked.' Below this, it says 'Enable Messaging API with the following info?'. It lists 'Account name: test' and 'Provider name: WebApplication'. At the bottom, there are 'Cancel' and 'OK' buttons.

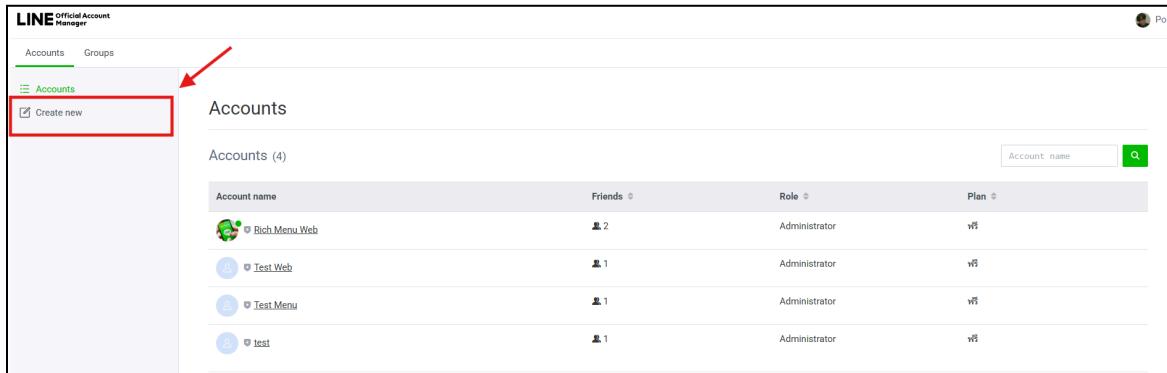
6. เรากำลังได้ Channel ID/ secret แล้ว

The screenshot shows the LINE Official Account Manager interface. The top navigation bar includes 'Home', 'Insights', 'Chats', 'Profile', 'LINE VOOM', and 'Extensions'. On the left, a sidebar titled 'Settings' contains sections for 'Account settings', 'Manage permissions', 'Response settings', 'Messaging API' (which is currently selected), 'Registered info', 'Activity and billing' (with 'Dashboard', 'Monthly plan', 'OA Chat package', 'Premium ID', 'Billing history', 'Payment method', and 'Invoice info' listed), and a link to the 'LINE Developers API documentation'.

The main content area is titled 'Messaging API'. It displays the status as 'Enabled'. Under 'Channel info', the 'Channel ID' is listed as '2007986265' with a 'Copy' button. Below it, the 'Channel secret' is listed as '4c000e65f104-97d3-472991676-9e14c' with another 'Copy' button. A 'Webhook URL' field contains 'https://'. At the bottom right of the form is a green 'Save' button. A note at the bottom states: 'You can find more related settings in the [LINE Developers Console](#)'.

- B) ยังไม่มี Line Official Account

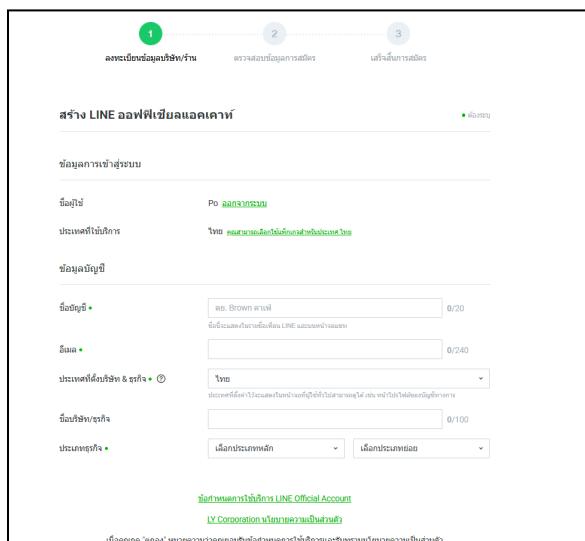
1. เข้าสู่ [LINE Official Account Manager](#) และเลือก Create new



The screenshot shows the 'LINE Official Account Manager' dashboard. The 'Accounts' tab is selected. In the sidebar, there is a 'Create new' button with a checked checkbox, which is highlighted with a red box and a red arrow pointing to it.

Account name	Friends	Role	Plan
Rich Menu Web	12	Administrator	ฟรี
Test Web	1	Administrator	ฟรี
Test Menu	1	Administrator	ฟรี
test	1	Administrator	ฟรี

2. กรอกรายละเอียดทั้งหมดให้ครบถ้วน



สร้าง LINE อย่างไรให้เข้าใจและคาดคะเน

ชื่อผู้ใช้: Po_ทดสอบระบบ
อีเมล: amcpozxc@gmail.com

ชื่อบัญชี:

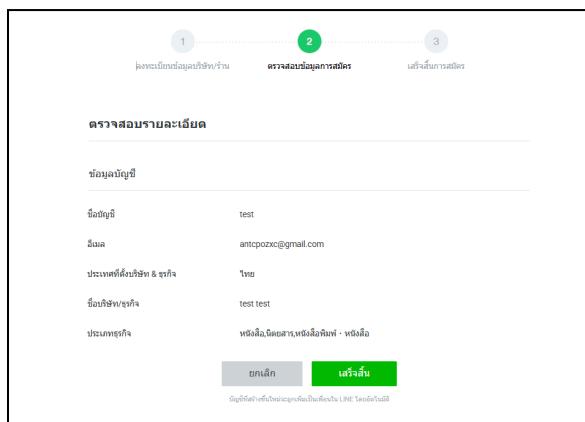
ชื่อผู้ใช้*: Po_Brown ลาบะ
อีเมล*: Po_Brown ลาบะ@gmail.com

ประเภทผู้ใช้งาน & สถานะ*: ไทย
ชื่อบัญชี/ชื่อร้าน: test test
ประเภทผู้ใช้งาน: บริษัท/องค์กร

เงื่อนไขการใช้งาน LINE Official Account
IY Corporation จำกัดขอสงวนสิทธิ์

เมือง: กรุงเทพฯ ประเทศ: ประเทศไทย

3. ตรวจสอบข้อมูลและกด “เสร็จสิ้น”

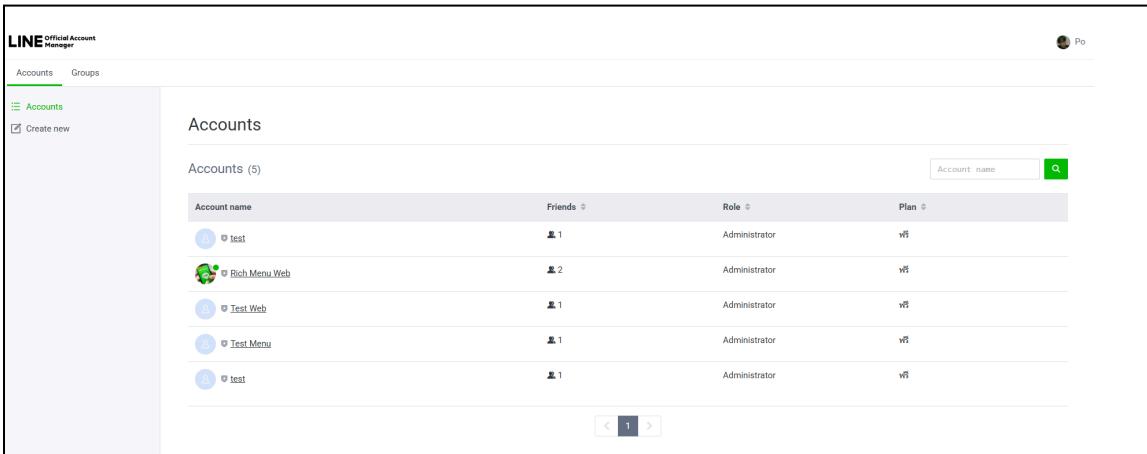


ตรวจสอบรายละเอียด

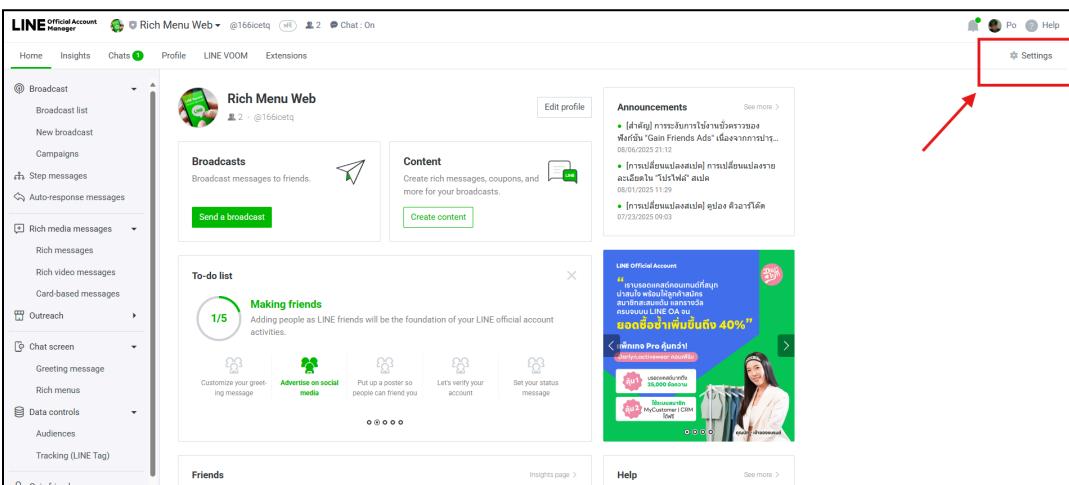
ชื่อบัญชี:
ชื่อผู้ใช้: test
อีเมล: amcpozxc@gmail.com
ประเภทผู้ใช้งาน & สถานะ: ไทย
ชื่อบัญชี/ชื่อร้าน: test test
ประเภทผู้ใช้งาน: บริษัท/องค์กร

ยกเลิก **เสร็จสิ้น**

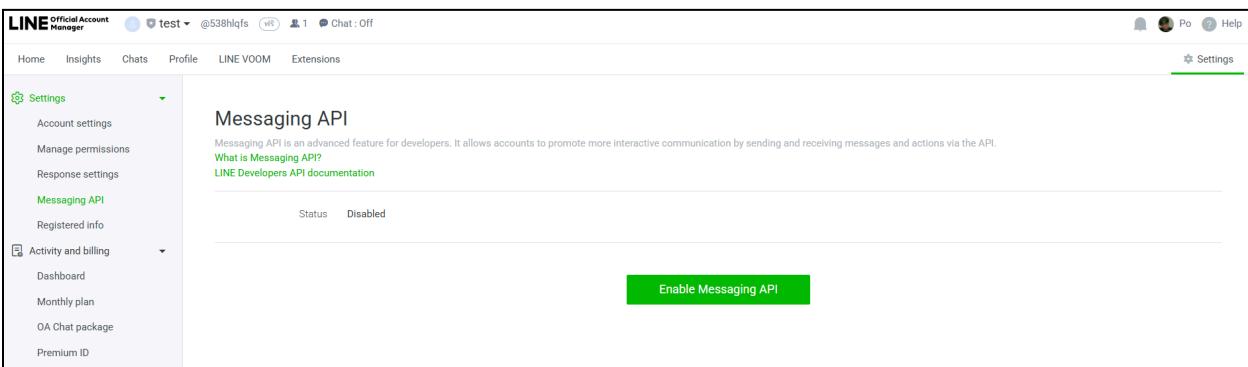
4. เมื่อเสร็จแล้วกลับมาเลือก Account ที่ต้องการจาก List



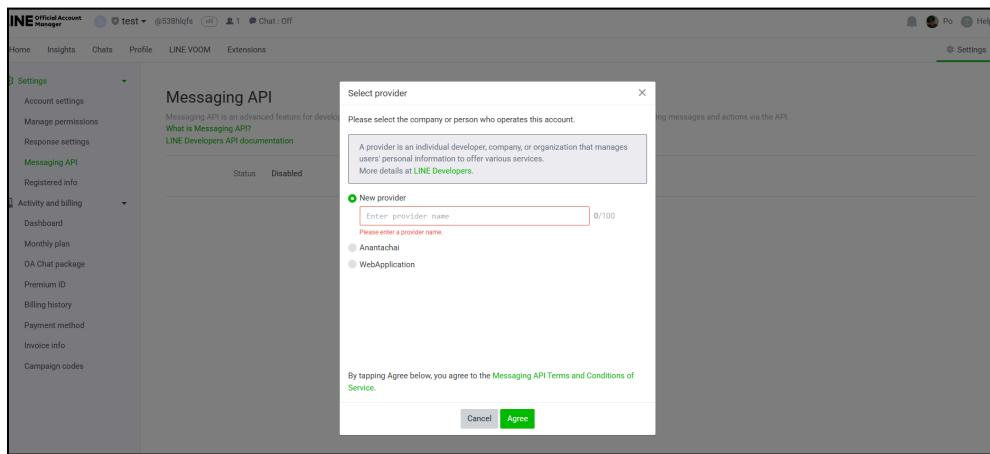
5. กดไปที่ setting หมุนบนขวา



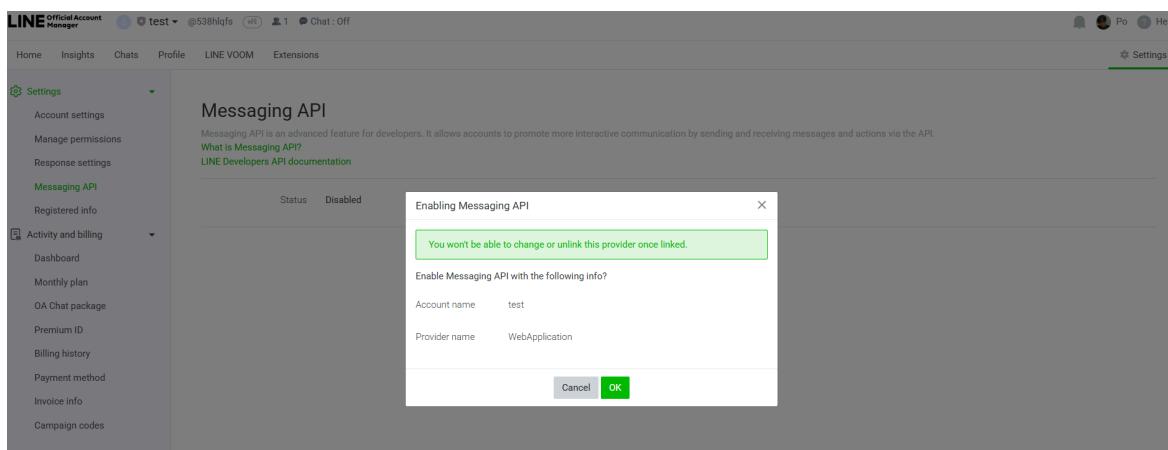
6. ไปที่หัวข้อ Messaging API และกด Enable Messaging API



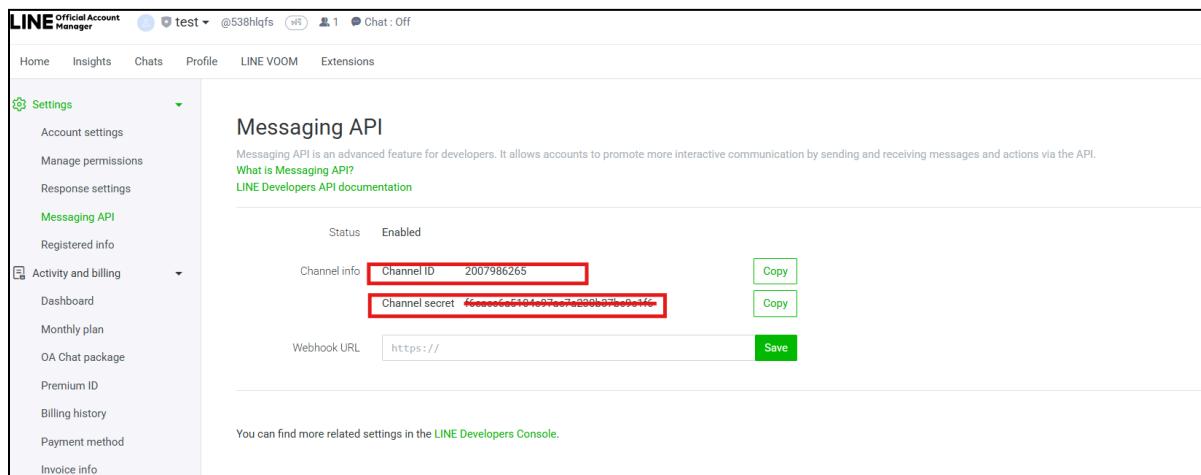
7. เลือก Provider ที่ต้องการ หรือสร้าง Provider ใหม่



8. กด OK

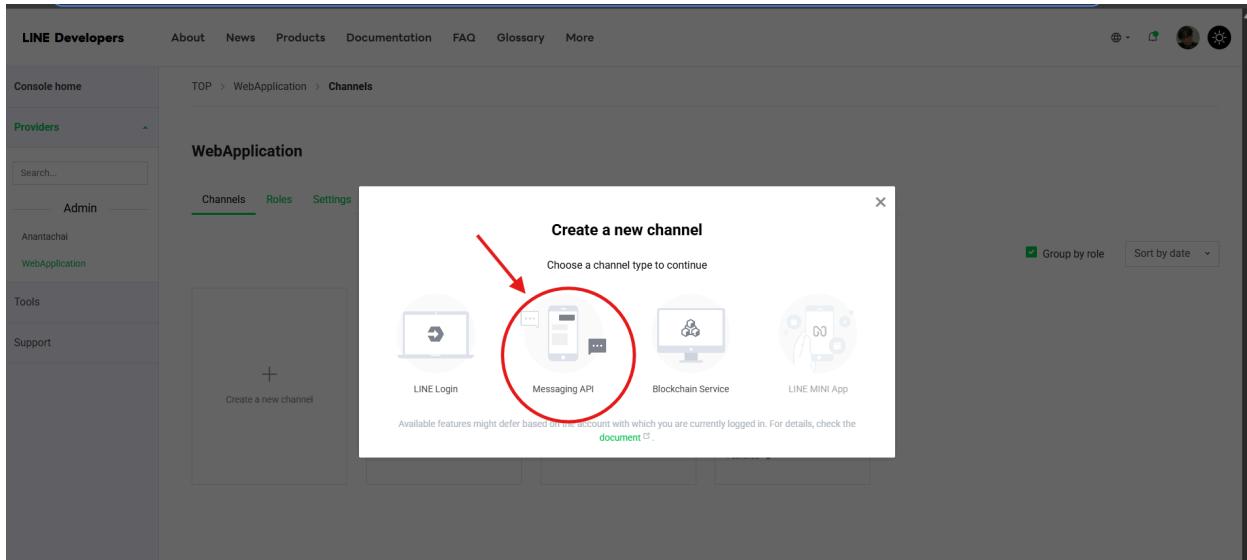


6. เรากำได้ Channel ID/ secret แล้ว

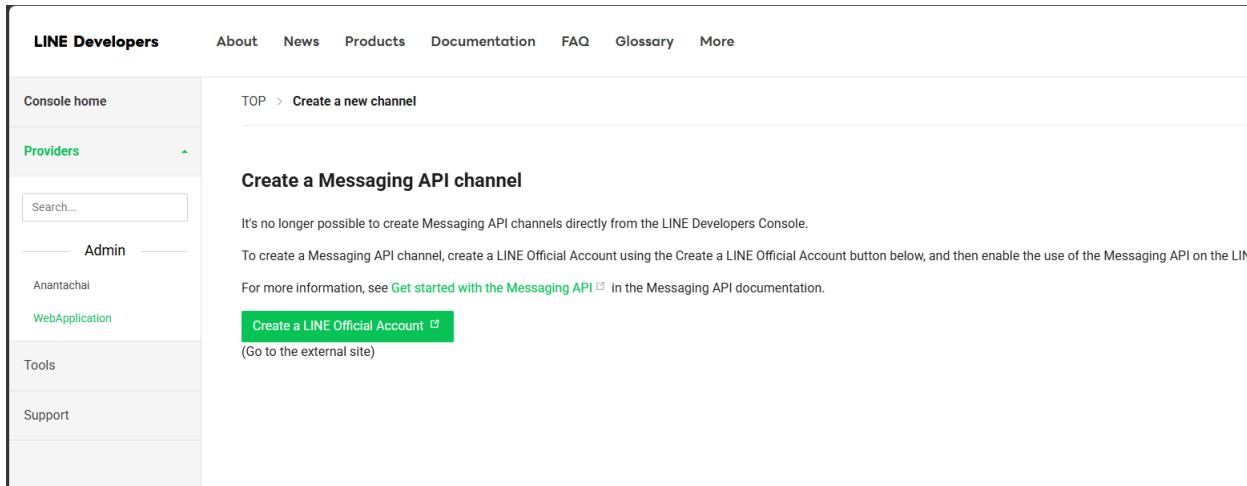


- C) สร้างจาก Line Developer Console

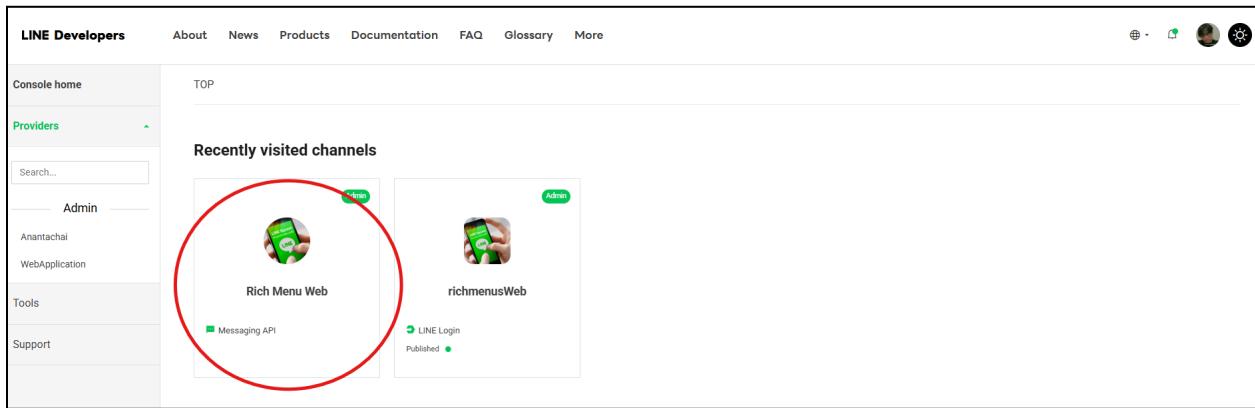
1. เข้า **LINE Developers Console** → เลือก **Provider** ของคุณ (หากยังไม่มี line official account สามารถดูสร้างได้ตามรูป)



2. เปิด Channel ที่เป็น **Messaging API** และกดเข้าไปสร้าง Line Official Account สามารถย้อนกลับไปดูที่ B) ต่อได้เลย

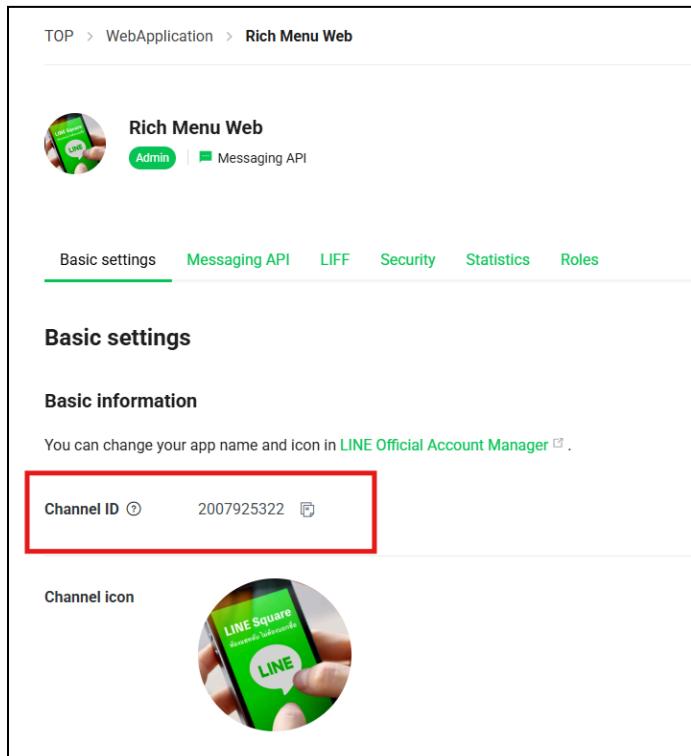


3. เลือก Channel ที่เป็น Messaging API จาก Line Official Account ที่สร้าง



The screenshot shows the LINE Developers console interface. On the left, there's a sidebar with 'Console home', 'Providers' (selected), 'Search...', 'Admin' (highlighted), 'Anantachai', 'WebApplication', 'Tools', and 'Support'. The main area is titled 'TOP' and shows 'Recently visited channels'. It lists two channels: 'Rich Menu Web' and 'richmenuWeb'. 'Rich Menu Web' is circled in red. Both channels have an 'Admin' status badge and a green 'Messaging API' badge.

4. เลือกแท็บ Basic settings: ดูค่า Channel ID



The screenshot shows the 'Basic settings' page for the 'Rich Menu Web' channel. At the top, it displays the channel name, icon, and status ('Admin | Messaging API'). Below this, there are tabs for 'Basic settings' (selected), 'Messaging API', 'LIFF', 'Security', 'Statistics', and 'Roles'. The 'Basic settings' section contains a 'Basic information' sub-section with a note about changing the app name and icon in the LINE Official Account Manager. It shows the 'Channel ID' field, which is '2007925322' and has a copy icon next to it. This entire section is highlighted with a red box. Below this is a 'Channel icon' section featuring a circular icon of a hand holding a smartphone displaying a green LINE interface.

5. แท็บ Messaging API: เลื่อนลงมาด้านล่างสุด ดูค่า Channel secret

The screenshot shows the Rich Menu Web admin interface. At the top, there's a profile icon of a hand holding a smartphone displaying a rich menu, followed by the text "Rich Menu Web". Below it, there are two buttons: "Admin" and "Messaging API". A navigation bar below these buttons includes tabs for "Basic settings", "Messaging API" (which is underlined in green, indicating it's active), "LIFF", "Security", "Statistics", and "Roles". The main content area is titled "Messaging API settings".

This screenshot shows the "Channel access token" section within the "Messaging API settings". It contains a long, redacted string of characters representing the channel access token. There is also a small "Copy" icon next to the redacted text.

ເຄົາມາໃຊ້ອ່າງໄຮ

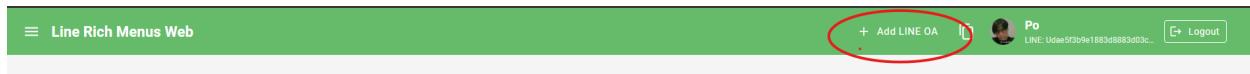
1. เปิดหน้า Accounts → กด Add LINE OA
2. กรอก Channel ID และ Channel secret แล้วกด เชื่อมต่อ
3. หากเชื่อม OA ที่เคยเชื่อมแล้ว ระบบจะ อัปเดตໂທເຄີນ/ຂ້ອມລຸລ່າສຸດ ให້າໄດຍໍໄມ່ສ້າງຫຼັງ

4) หน้าจอ Accounts

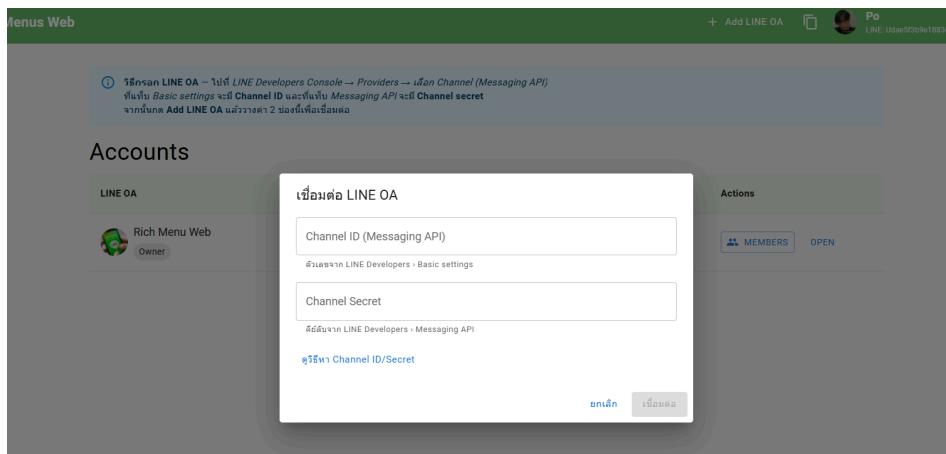
ใช้สำหรับ เพิ่ม/ลบ OA ที่ผูกกับบัญชีเรา และ จัดการสมาชิก ของแต่ละ OA

เพิ่ม OA (Add LINE OA)

1. กดปุ่ม Add LINE OA มุมขวาบน ที่หน้า



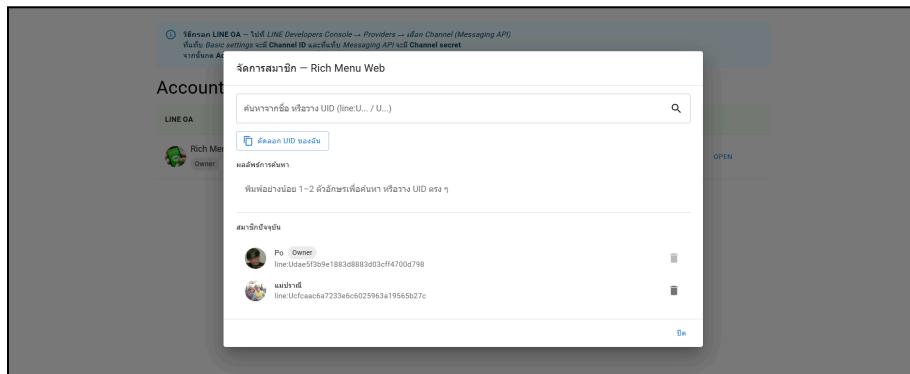
2. กรอก Channel ID และ Channel secret จาก LINE Developers



3. กด เชื่อมต่อ เพื่อเพิ่ม OA เข้ารายการ

จัดการสมาชิก (Members)

- เฉพาะ Owner ของ OA นั้น ๆ ที่เพิ่ม/ลบสมาชิกได้
- ค้นหาจากชื่อ หรือว่าง UID ตรง ๆ ได้ (รูปแบบ `line:Uxxxxxxxxx...` หรือ `Uxxxxxxxxx...`)



5) หน้าจอ Broadcast

ส่งข้อความถึงเพื่อนทั้งหมดหรือทำ Targeting พร้อมตั้งเวลาได้

ขั้นตอน (Workflow)

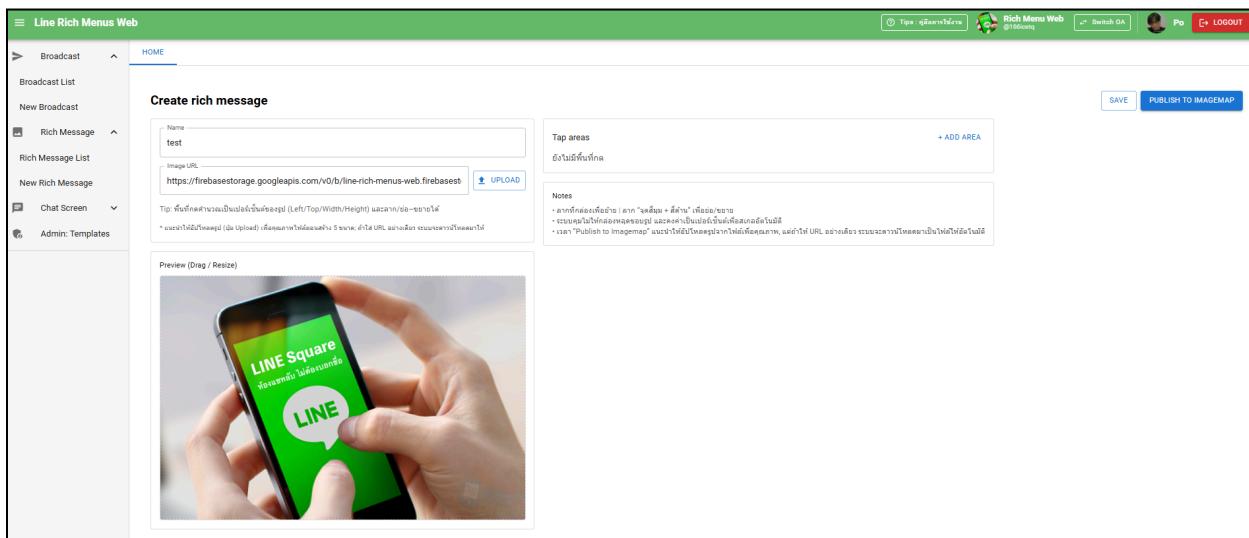
The screenshot shows the 'Broadcast' section of the Line Rich Menus Web interface. On the left, there's a sidebar with options like 'Broadcast List', 'New Broadcast', 'Rich Message', 'Chat Screen', and 'Admin: Templates'. The main area is titled 'Broadcast' and includes fields for 'Recipients' (set to 'All friends'), 'Broadcast time' (set to 'Send now'), and a large text input field with a placeholder 'Enter text' and an emoji icon. There are also buttons for 'SAVE DRAFT', 'SEND TEST', and 'SEND'.

1. เลือกผู้รับ: **All friends**
2. เวลา: **Send now** หรือ **Schedule** (ระบุ Date/Time/Timezone)
3. แต่งคอนเทนต์ด้วย **Blocks**: Text / Image / File / Link / Rich (เลือก Rich Message/Imagemap ที่สร้างไว้)
4. เลือก **Send test**, **Save draft** หรือ **Send**

6) หน้าจอ Rich Message

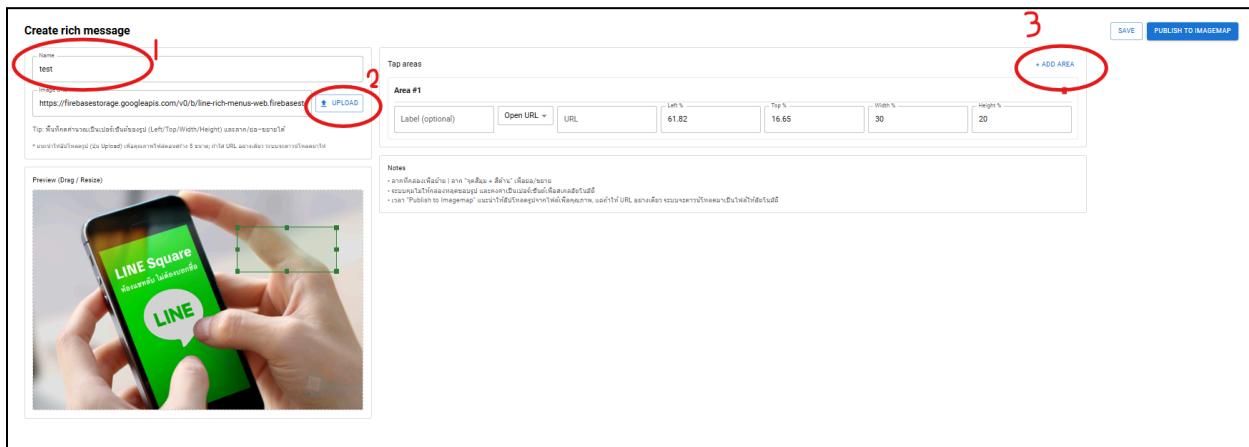
สร้างภาพ + พื้นที่กด (Imagemap) เพื่อใช้ข้าใน Broadcast

ขั้นตอน



1. ไปหน้า Rich Message → กด Create

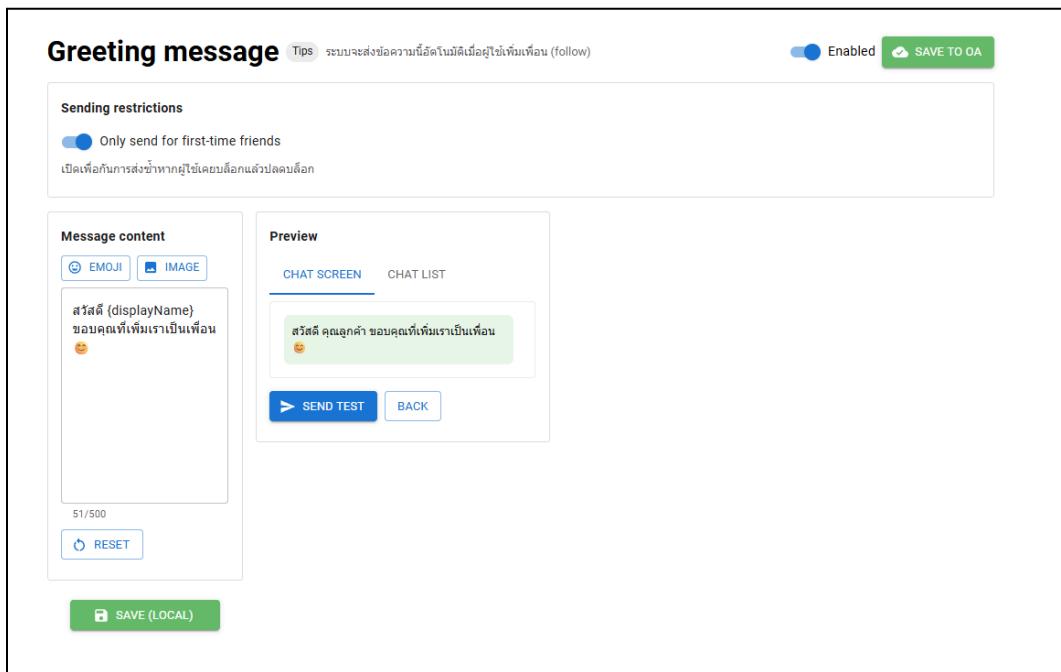
2. ตั้งชื่อ, อปปโนลดรูป และเพิ่ม areas (label + URL)



3. บันทึก (โหมด Guest เก็บในเครื่อง, โหมด Login บันทึกจริง) และเลือกไปใช้ใน Broadcast

7) หน้าจอ Greeting Message

ข้อความต้อนรับเมื่อผู้ใช้ เพิ่มเพื่อน OA ของคุณ



การตั้งค่า

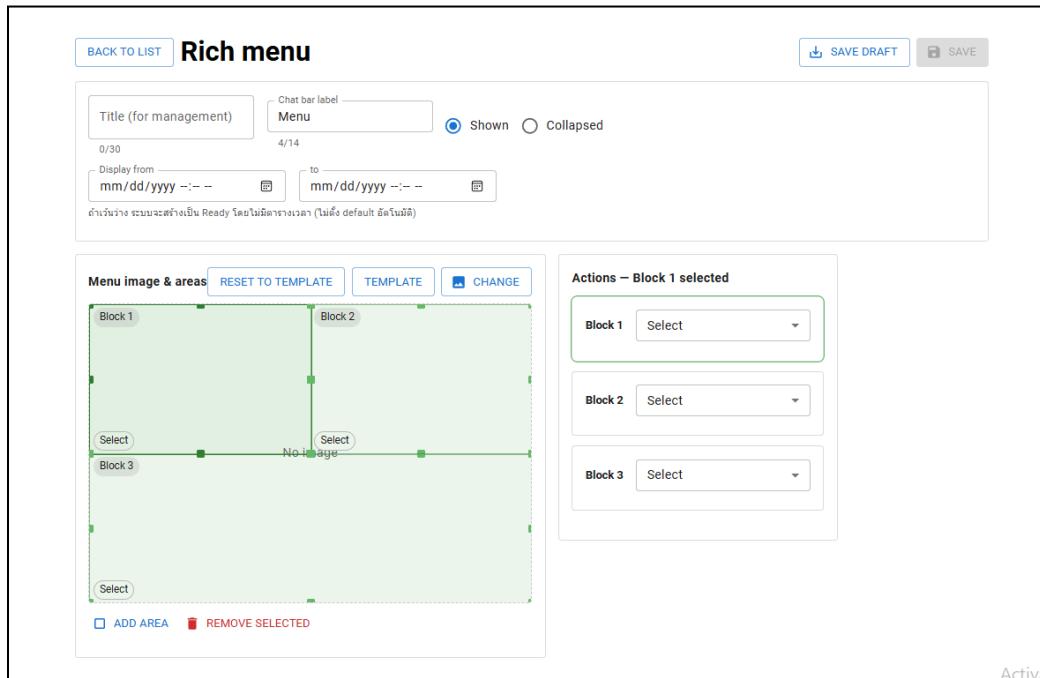
- เปิด/ปิดการใช้งาน, เลือกส่งเฉพาะครั้งแรกที่เพิ่มเพื่อน
- ตั้ง ข้อความ + รูปภาพ (รองรับตัวแปร {displayName}, {accountName})
- บัน Save, Save to OA, หรือ Send test

ตัวอย่างข้อความ

1. สวัสดี {displayName} ขอบคุณที่เพิ่ม {accountName} 😊
2. พิมพ์ "เมนู" เพื่อดูสิทธิพิเศษวันนี้ได้เลย!

8) หน้าจอ Rich Menu (+ Action QnA)

ออกแบบเมนูภาพที่คลิกได้ เลือกเทมเพลต อัปโหลดรูปตามสเปค LINE และกำหนด Action ให้แต่ละบล็อก



สเปคภาพ (แนะนำ)

- **Large:** 2500×1686 px
- **Compact:** 2500×843 px
- ขนาดไฟล์ควร $\leq \sim 1$ MB, Chat bar label ≤ 14 ตัวอักษร

ตัวอย่างเทมเพลต

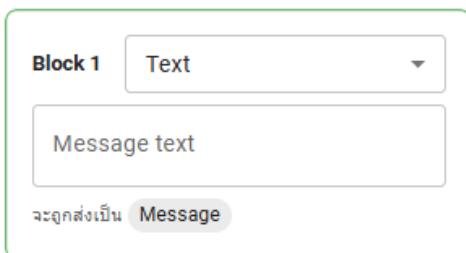
- Large 6 บล็อก ($2 \times 2 \times 6$)
- Large 3 บล็อก (3×2 , 3×2 , 6×2)
- Compact 4 บล็อก ($3 \times 2 \times 4$)
- Compact เดิมทั้งผืน (1 บล็อก)

Action ที่รองรับ

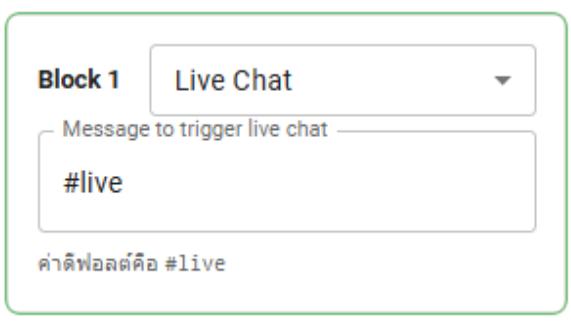
- **Link:** เปิดหน้าเว็บ (URL) — ใช้กับโปรดิวชัน/คุปอง/จองโต๊ะฯลฯ



- **Text:** ส่งข้อความทันที — เช่น “ดูเมนูวันนี้” เพื่อให้บอทตอบต่อ



- **Live Chat:** ตั้งข้อความทวิกเกอร์ เช่น #live เพื่อสั่งเป็นโหมดแชทสด



- **QnA (สำคัญ):** ส่ง **postback** รูปแบบ **qna :<key>** เพื่อดึงค่าตอบ FAQ อัตโนมัติ

Actions – Block 1 selected

Block 1 QnA

หมวดคำถาม (QnA)

ข้อความที่จะแสดงในแชท (กดแล้วส่ง)

ข้อความตอบกลับสำรอง (ถ้าไม่มีคำตอบ)

จะส่งเป็น **postback** data: qna: ...

รายการคำถามยอดเยี่ยด **+ ADD QUESTION**

#1 REMOVE

Question 1

Answer

ผู้ใช้สามารถพิมพ์หมายเลข 1–1 หรือพิมพ์คำถาม/ค่าหลักที่ “คล้าย” กับรายการ เพื่อรับคำตอบ

9) หน้าจอ Template Rich Menus (สำหรับแอดมิน)

- ใช้สร้าง/แก้ไข เทมเพลต Rich Menu กลาง ที่ทีมสามารถเลือกไปใช้งานต่อได้
- การอัปโหลดรูปจะเก็บที่ Firebase Storage ภายใต้โฟลเดอร์ตามที่กำหนด
- ลิฟท์ชีฟีลด์ที่ต้องระบุเมื่อเพิ่ม Rich Menu ใหม่ (ดูข้อ 10)

10) หน้า Task Assignment

The screenshot shows the Line Rich Menus Web interface. On the left, there's a sidebar with options like Broadcast, Rich Message, Chat Screen, and Task Assignment (which is selected and highlighted with a red arrow). The main content area is titled "Task Assignment". It contains a sub-section "การรวมการทำงาน" (Integration Work) which explains how to integrate LINE with Google Sheets and OA. It includes steps for setting up LINE Webhook URL and Google Sheet ID. A "VERIFY CONNECTION" button is present. At the bottom right, there's a message "Activate Windows Go to Settings to activate Windows.".

ภาพรวมการทำงาน

บอทช่วย “สั่งงาน-ติดตามงาน” ผ่าน LINE โดยบันทึกงานลง Google Sheets ของ OA นี้โดยตรง ผู้ใช้พิมพ์คำสั่งภาษาคุณ (เช่น “@po ทำรายงาน พรุ่งนี้ 17:30”) ระบบจะสร้างงาน กำหนดผู้รับผิดชอบ เดดไลน์ และแจ้งเตือนล่วงไปกว่าเวลาระยะเวลาที่กำหนด

ประโยชน์ของการใช้งาน

- ลดความวุ่นวายในการตามงาน (มีสถานะ/กำหนดส่งขัดเจน)
- ใช้งานง่ายผ่านแชท
- ข้อมูลเป็นของ OA นี้ 1:1 ในชีท
- รองรับบทบาท/สิทธิ์
- เปิด/ปิดและปรับค่า Rich menu ได้จากหน้านี้

ตั้งค่า LINE Webhook :

① ตั้งค่า LINE Webhook

ตั้งค่า Webhook URL ใน LINE Official Account ของคุณเป็น:

`https://line-rich-menu-web.onrender.com/webhook/line`

* ไปที่ Messaging API → กรอก Webhook URL และ กดปุ่ม save

* ไปที่ Response settings → Webhook settings และ เปิด (Enable) WebHook and Chat

กดปุ่ม Verify Connection เพื่อตรวจสอบการเชื่อมต่อ

* เนื่องในเวอร์ชันของรุ่น URL เดียวสำหรับทุก OA: /webhook/line

กด Enable เพื่อเปิดการใช้งาน Webhook ของ Task assignment



การตั้งค่า/และวิธีการหาค่า Google Sheet ID

Google Sheet (1:1 ต่อ OA)

Google Sheet ID (ของ OA นี้) — 1LWyHKMktEOMKnwR5O-1hXuzVlVRIA5wvCTqQ9nPkc

คัดลอก ID ระหว่าง /d/ และ /edit จากลิงก์ของสเปรดชีต

VERIFY CONNECTION ล่าสุด: 9/26/2025, 1:08:47 PM

วิธีหาค่า Google Sheet ID
ด้วยวิธีการตามที่แนะนำ

คลิกที่รูปเพื่อดูรายละเอียด

- 1) เปิดสเปรดชีตใน Google Sheets และอุปกรณ์ที่ URL เช่น <https://docs.google.com/spreadsheets/d/1AbCDef...XYZ/edit#gid=0>
- 2) คัดลอกเฉพาะชื่อความเร็ว /d กับ /edit
- 3) วางแผนในช่อง "Google Sheet ID" และกด Save
- 4) กด Verify เพื่อตรวจสอบว่าเชื่อมต่อ
หมายเหตุ: ตั้งค่า Share Google Sheet เมื่อ Anyone with the link = "Editor"

SAVE

ปุ่ม “VERIFY CONNECTION”

VERIFY CONNECTION

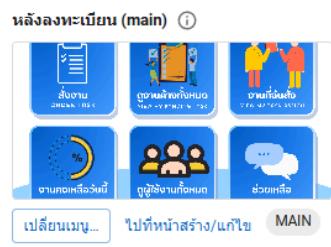
ล่าสุด: 9/26/2025, 1:08:47 PM

ใช้สำหรับตรวจสอบการเชื่อมต่อของ Apps Script

ส่วนที่แสดง Rich Menu ที่ใช้งานเมื่อ กด “Enable”

≡ Rich menu สำหรับ OA นี้

เลือกเมนู ก่อนลงทะเบียน (prereg) และ หลังลงทะเบียน (main) — เมื่อกด Enable ระบบจะตั้งค่าให้ OA โดยอัตโนมัติ



SAVE & APPLY

ส่วนแนะนำและวิธีการใช้งานคำสั่งต่างๆในระบบ

① บทบาทที่รองรับ & ลิสต์การใช้งาน

developer – นักพัฒนา admin – ผู้ดูแล supervisor – หัวหน้างาน user – ผู้ใช้งาน

สรุปสิทธิ์:

- ผู้ใช้งาน (สั่งงานในหน้า): ทุกบทบาทท่านได้ (พิมพ์ในช่อง → พิริว → บันทึก)
- ผู้ดูแล (สถานะ/เดดไลน์/รายละเอียด/โน๊ต/เตือน): เจ้าของงาน (คนที่สั่ง) หรือบทบาท admin/supervisor/developer
- * ลิสต์สิทธิ์: developer > admin > supervisor > user

๔. วิธีใช้งาน & คำสั่งหลัก

เริ่มต้น / ลงทะเบียน

- พิมพ์ ลงทะเบียน เพื่อเปิดฟอร์มครึ่งแรก
- กรอก ชื่อจริง, ชื่อเล่น (username), ลำดับที่ หรือบุคคล LINE ของคุณ
- ดำเนินการโดยแตะแล้ว ระบบจะแจ้งข้อมูลปีจุบัน และมีล่วงเลือก แก้ไขข้อมูล



สั่งงาน (พิมพ์ในช่อง)

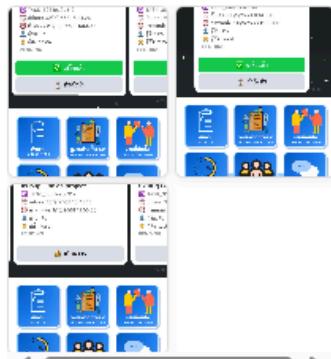
- @po ปรับรายการ พัชร์นี้ 09:00
- @test ขอทำปีบาน้ำร้าน ก่อนม่าย 3 (ดึงความเวลาปั้น 15:00 ของวันนี้)
- @po ไฟ rich menu วันนี้ ล่วน (ดึงแท็ก [URGENT])

เหลือลิส: ไม่ได้เวลา — ไฟ 17:30 ล่อใจมี, ค่าว่า "ก้อนน้ำ 3" — 15:00 ร้าน



ดูงานของฉัน / อัปเดตงาน

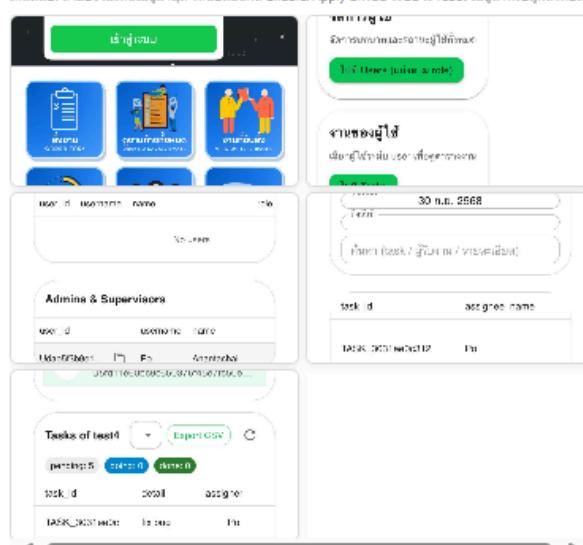
- ดูงานที่กำกับหมวด – งานสถานะ pending/doing
- งานของฉันเป็นนี้ – งานที่ค้างงานส่ง “วันนี้”
- งานที่สั่งสื้น – งานที่สั่งเป็นผู้สั่ง
- เสร็จงาน: done <TASK_ID>
- ค้างส่งท้า: ค้างส่งต่อในภารกิจ <TASK_ID>
- แก้ค้างงานดัง: แก้ค้างงานดัง <TASK_ID> พุธนี้ 10:00
- เที่ยวน้อง: เที่ยวน้อง <TASK_ID> รอความ...



หน้าเริ่ม “จัดการผู้ใช้งาน”

- ในแบบ พิมพ์ จัดการผู้ใช้งาน → มอบสิทธิ์ที่มีปุ่ม “เข้าสู่ระบบ”
- คลิกปุ่ม → เปิดเป็นตัวอย่าง magic link (อายุ ~2 ชม.) โดยไม่ต้องกรอกรหัสผ่าน
- หัวข้อ Home และส่ง “รหัสผู้ใช้ @username” และหมายเหตุของคุณ
- ไปที่ Users (ແນ່ດາມ role) เพื่อจัดการสิทธิ์/รายชื่อ

เหลือลิส: ลืมรหัสผู้ใช้บัญชีลูก ໃบ้และมีบุก Enable/Apply มีการ รับบัน reset บัญชีหัวหน้าผู้ดูแลโดยอัตโนมัติ



สรุปคำสั่งทั้งหมด (Cheat-Sheet)

[ลงทะเบียน / 注销ไฟล์]

- ลงทะเบียน
- ลงทะเบียน <username> <ชื่อจริง> <role>
 - ล็อกอินลงทะเบียนแล้ว พิมพ์ "ลงทะเบียน" อีกครึ่งเพื่อคุณจะเข้าบัน
- จัดการผู้ใช้งาน
 - + เปิดการ์ดเพื่อเปลี่ยนรหัสบันทึก (magic link)
- ตัดตอนเดือน
 - + ด้วยอย่าง: dm @po ของสีที่แนบท้าย

[สังงาน (มีพิธีวิ่งก่อนเขียนอีก)]

- @<username> รายงานอีเมลงาน [เปลี่ยนราย/โนดแบบภาษาคน]
 - ↳ นาฬิกาส่งหรือว่างงานข้าวคลาวเป็นรหัส TMP_xxx

• อีเมลบนหน้าจอ TMP_xxx

• ยกเลิกบนหน้าจอ TMP_xxx

ด้วยอย่าง:

- @po ทำรายงาน พุ่งนี้ 09:00
- @test ทำเป้าหมายร้าน ก่อนน้ำย 3
- @po ทำ rich menu วันนี้ คุณ

[ด้วยอย่างภาษาคน (เวลาที่ระบบเข้าใจ)]

- “วันนี้ 17:30” (ถ้าไม่ใส่เวลา ระบบจะค่าเริ่มต้น 17:30)
- “พุ่งนี้ 09:00”
- “ก่อนน้ำย 3” → 15:00 ของวันนี้
- “วันรุ่นที่ 10:00” (เมื่อวันไทย)
- ค่าวา “คุณ” / “urgent” จะติดแท็ก [URGENT] ให้สื่อใจว่า

[ดูงาน / ค้นหารายการของฉัน]

- ดูงานค้างทั้งหมด → งานสถานะ pending/doing
- งานของฉันวันนี้ → งานที่กำหนดลง “วันนี้”
- ดูงานที่สั่นสั่ง → งานที่เราเป็นผู้สั่ง (assignee)

[ปรับเด้งงาน]

- done <TASK_ID> → มีงาน
- กำลังดำเนินการ <TASK_ID> → ห้องสถานะ doing
- แก้กำหนดลง <TASK_ID> <เวลา/ภาษาคน>
- เพิ่มนัด <TASK_ID> <ข้อความ>

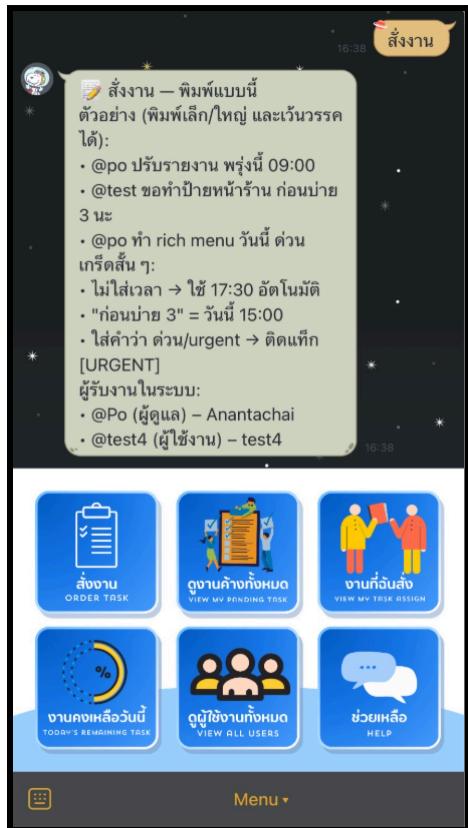
[สิทธิ์โดยสิ้นเชิง]

- ทุกบทบาทสามารถ “ดูงาน” ได้
- การแก้ไขงาน (สถานะ/เดดไลน์/รายงานอีเมล/โนด): ทำได้โดย
 - เจ้าของงาน (คนที่สั่ง) พร้อม
 - หน้าท admin / supervisor / developer
- ลำดับสิทธิ์: developer > admin > supervisor > user

ลงทะเบียน



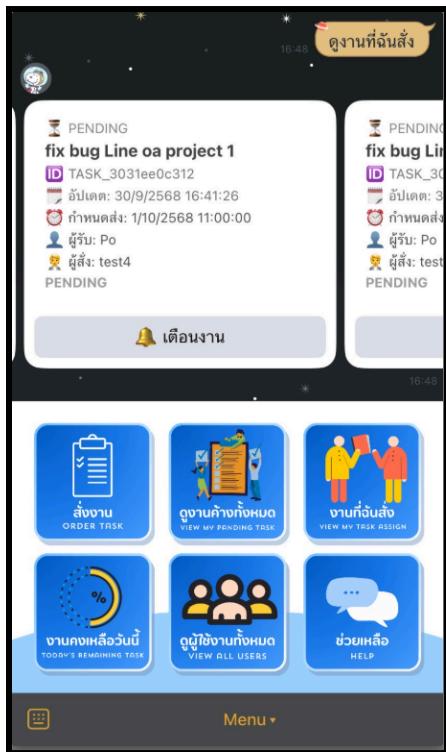
ขั้งงาน



ดูงานค้างทั้งหมด



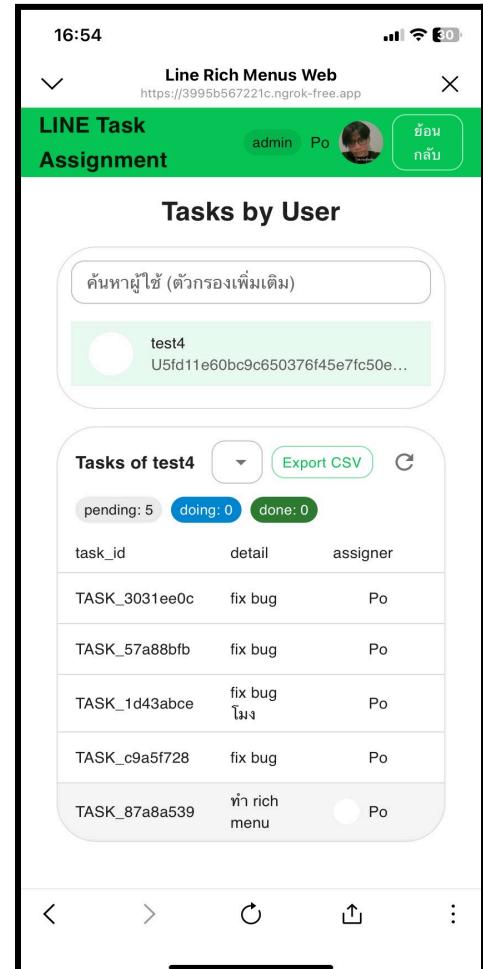
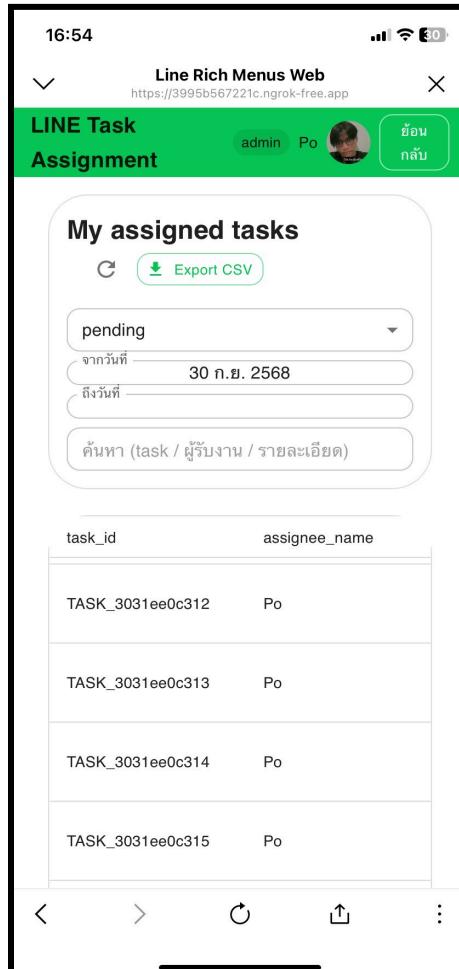
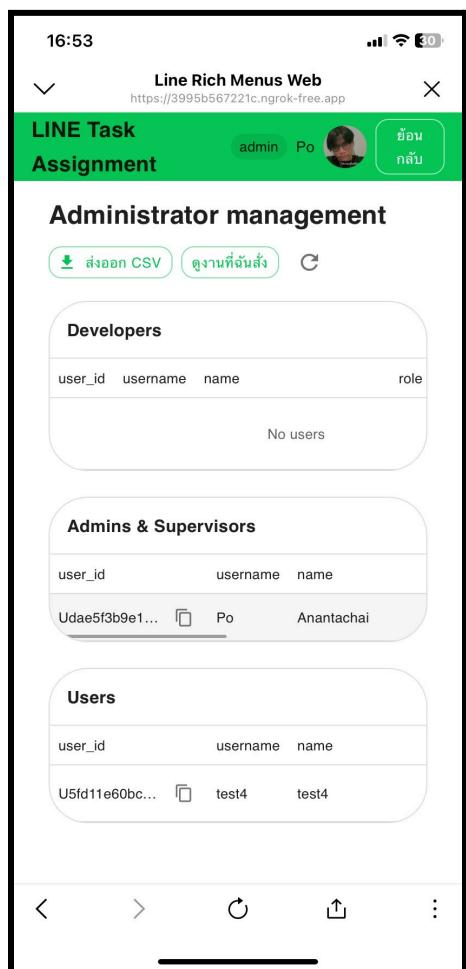
ดูงานที่ฉันสั่ง



งานของฉันวันนี้



หน้าเว็บ “จัดการผู้ใช้งาน”



11) หน้า Time Attendance

The screenshot shows the 'Line Rich Menus Web' interface with the 'HOME' tab selected. On the left, there's a sidebar with options like Broadcast, Rich Message, Chat Screen, and Template Line OA. Under 'Template Line OA', it says 'Task Assignment' and 'Time Attendance'. The main area is titled 'Time Attendance' and contains the following sections:

- การรวมการห่างงาน**: ระบุ URL ของ Google Sheets และ OA ที่จะบันทึกข้อมูลใน Google Sheets และ OA นั้นๆ และตั้ง Rich menu แบบห่างงาน Admin กับ User
- Status**: Disabled (ปิด)
Description: เลื่อนใช้การบีบอัดข้อมูล เช่น Save Google Sheet ID และ Verify ลักษณะ
- ผู้ต่อ LINE Webhook**:
เลือก LINE Webhook URL ที่ต้องการ
URL: <https://lineeo.superhr.biz/webhook/line>
Description: 1) กรอก URL และกด Save • 2) ไป Webhook ตรวจสอบ • 3) กดมาหน้านี้แล้ว Verify Connection
- Google Sheet (1:1 หรือ OA)**:
Google Sheet ID (was OA ด้วย)
ID: 1QEYvalMm6WGKujlgw70DNyVREwy7w-PczG9BkYkisE
Description: ต้องมี ID ของ Google Sheet และกด Verify Connection
- VERIFY CONNECTION**: วันที่: 11/26/2025, 9:10:17 PM
- SAVE**

At the bottom right, there's a message: "Activate Windows Go to Settings to activate Windows."

หน้า Time Attendance

เงื่อนไขเปิดใช้งาน กรอก sheet id ตามคำแนะนำ และ กด save & verify และกด enable เพื่อใช้งาน

The screenshot shows the 'Rich menu สำหรับ OA นี้' configuration screen. It has two main sections: 'Admin (owner/admin)' and 'User (ห้องงานทั่วไป)'.

Admin (owner/admin) (Left):
Icon: สองคนทำงาน
Buttons:

- เปลี่ยนเมนู...
- ไปที่หน้าสร้าง/แก้ไข
- ใช้เมนู DEFAULT
- ใช้ preset (Admin)

User (ห้องงานทั่วไป) (Right):
Icon: สามคนทำงาน
Buttons:

- เปลี่ยนเมนู...
- ไปที่หน้าสร้าง/แก้ไข
- ใช้เมนู DEFAULT
- ใช้ preset (User)

A large blue 'SAVE' button is located at the bottom center.

User Guide (คู่มือการใช้งาน user)

ฟัง User

1. ลงทะเบียนเข้าใช้งาน

- พิมพ์คำสั่ง “ลงทะเบียนเข้าใช้งาน” → กรอกข้อมูลส่วนตัว/บัญชีธนาคาร
- ระบบผูก LINE UserId กับโปรไฟล์ในชีต (employees) เพื่อใช้งานค่าสั่งอื่น



คำสั่ง “ลงทะเบียนเข้าใช้งาน”

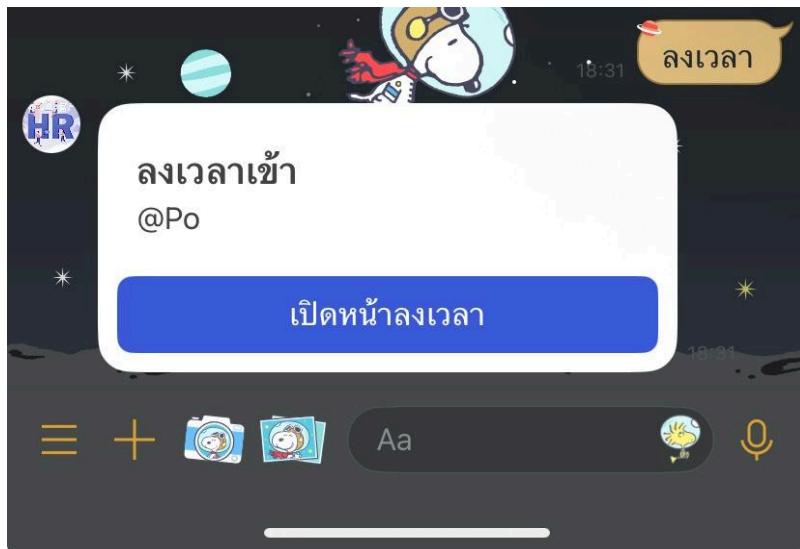


หน้าลงทะเบียนเข้าใช้งาน สามารถ
ทะเบียนด้วยการแสกน บัตรประชาชน

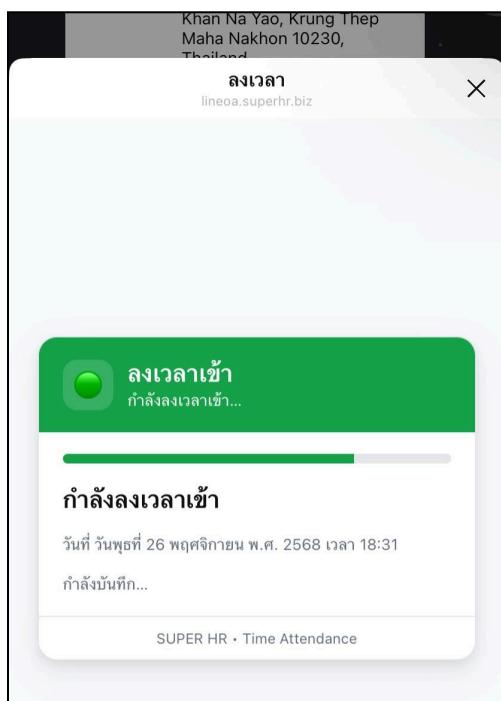
หมายเหตุ : การลงทะเบียนข้าจะเป็นการ
อัพเดทข้อมูลเดิม

2. ลงเวลา (เข้างาน)

- พิมพ์ “ลงเวลาเข้างาน” → ระบบบันทึกวัน–เวลา–พิกัด/ที่อยู่ (ถ้ามี) ไว้ใน work_logs_YYYY-MM
- ถ้ากดเข้าในวันเดียวกัน ระบบจะอัปเดต IN ของวันนั้นแทนการสร้างແควใหม่



คำสั่ง “ลงเวลาเข้างาน”



หน้าบันทึกข้อมูลลงเวลาเข้างาน

3. ออกงาน

- พิมพ์ “ลงเวลาออกงาน” → ระบบบันทึกวัน–เวลา–พิกัด OUT และลิงก์กับ IN ของวันเดียวกัน



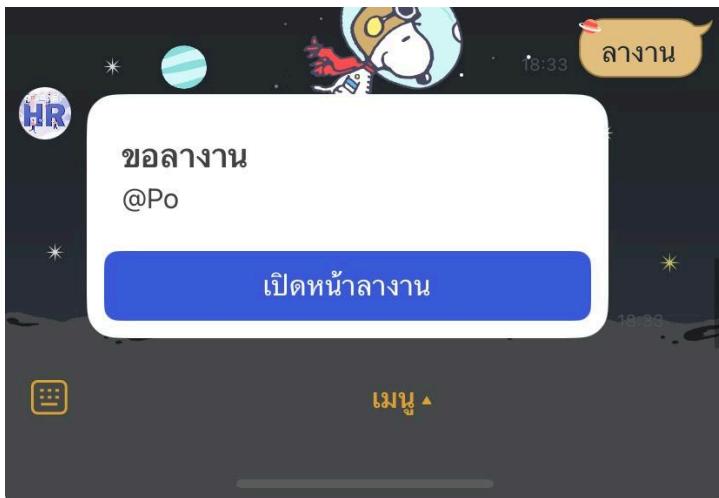
คำสั่ง “ลงเวลาออกงาน”



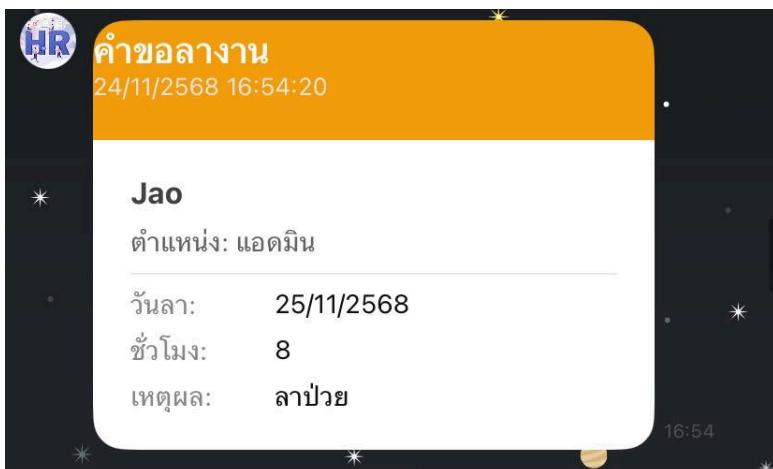
หน้าบันทึกข้อมูลลงเวลาออกงาน

4. ขอลา

- พิมพ์ “ลางาน” → ระบุวันที่/ชั่วโมง/เหตุผล
- ระบบคำนวณโควตาชั่วโมงลา/วันลา ตาม dailyHours (จากกะทำงาน – เวลาพัก) และ leaveQuotaDays



คำสั่ง “ลางาน”



Flex message ลางาน

5. ចំណាំលើខ្លួន

- ផ្តល់ព័ត៌មានអត្ថបទសំណង់ដែលត្រូវបានបង្ហាញ



ចំណាំលើខ្លួន

ผู้ Owner/Admin

1. บันทึกการทำงาน

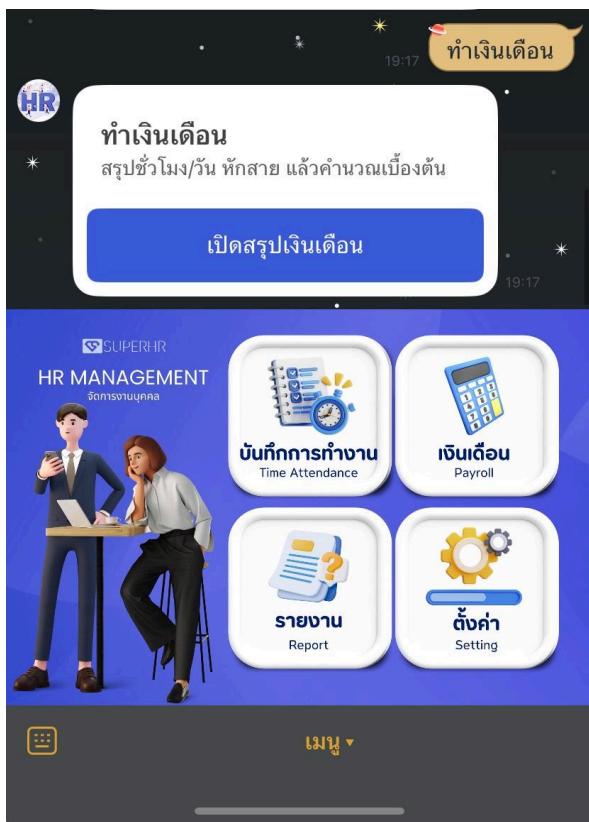
- ตรวจสอบ IN/OUT รายวันของพนักงานจากช่วงวันที่เลือก (รวมหลายชีต work_logs_YYYY-MM-YY)
- เห็นชั่วโมงดิบ (OUT-IN), หักพัก (breakMinutes) → ได้ “ชั่วโมงทำงานจริง” และ “ชั่วโมงเกินต่อวัน”

คำสั่ง “บันทึกการทำงาน”

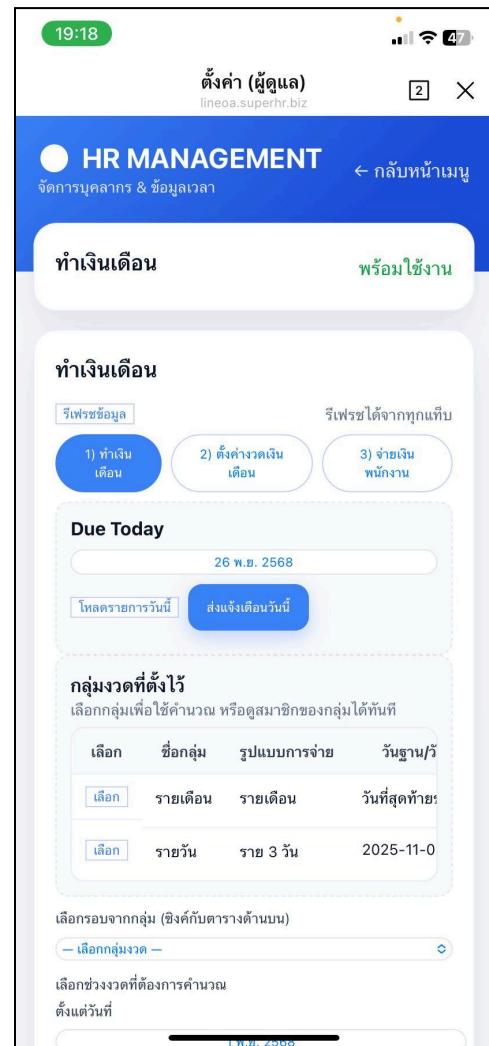
หน้าบันทึกการทำงาน

2. ทำเงินเดือน

- เลือกช่วงงวด (period) หรือใช้กลุ่มงวดเงินเดือน (Pay Groups) เพื่อคำนวณยอด
- ระบบรองรับ payType: hourly/daily/monthly/every_n_days/every_n_hours และตัวเลือก prorateLate
- ปัดเศษยอดเงินตามเกติกา (.50 ขึ้นไปปัดขึ้น) ก่อนบันทึกเป็นรายการจ่ายต่อคน
- สามารถปรับแก้ยอด/รายละเอียดแบบรายบุคคลแล้วบันทึกสถานะ (approved/paid ฯลฯ)



ตำแหน่ง “ทำเงินเดือน”



หน้าทำเงินเดือน 1

- หน้าทำเงินเดือน 1 คำนวนเงินตามงวด วันที่เลือก มีกลุ่มให้เลือกและ กดทำเงินเดือน จะบันทึกงวดนั้น

The screenshots show the following steps:

- Step 1: Wage Run Creation** (Left Screenshot)
- Step 2: Pay Run Selection** (Middle Screenshot)
- Step 3: Payment Confirmation** (Right Screenshot)

Step 1: Wage Run Creation

显示了工资发放的主界面，包含以下元素：

- 顶部状态栏显示时间为 19:20。
- 标题为 “HR MANAGEMENT”。
- 子标题为 “จัดการบุคลากร & ข้อมูลเวลา”。
- 主要操作按钮为 “ทำเงินเดือน” 和 “รีเฟรชแล้ว”。
- 下方显示了 “ทำเงินเดือน” 按钮，旁边有三个子选项：1) ทำเงินเดือน, 2) ตั้งค่าจ้างเงินเดือน, 3) จ่ายเงินพนักงาน。
- 下方显示了 “ตั้งค่า (ผู้ดูแล)” 和 URL “lineoa.superhr.biz”。
- 底部显示了工资发放的表格，列出了员工姓名、日期范围、金额等信息。
- 底部还显示了银行转账信息：“แก้ไขกลุ่ม: รายเดือน” 和 “บันทึก”、“ยกเลิก” 按钮。

Step 2: Pay Run Selection

显示了选择工资发放周期的界面：

- 显示了 “ทำเงินเดือน” 按钮和 “รีเฟรชแล้ว” 按钮。
- 显示了 “จ่ายเงินพนักงาน” 按钮。
- 显示了 “เลือกเดือน” 按钮，下方显示了 “พฤศจิกายน 2568”。
- 显示了 “เลือก Pay Run” 按钮，下方显示了 “PAY-2 · 2025-11-01 → 2025-11-30”。
- 显示了 “รอจ่าย (Approved)” 表格，列出了金额 14,823.96 和 1,501.5，以及 “จ่ายเงินพนักงาน” 按钮。

Step 3: Payment Confirmation

显示了支付确认界面：

- 显示了 “ทำเงินเดือน” 按钮和 “รีเฟรชแล้ว” 按钮。
- 显示了 “จ่ายเงินพนักงาน” 按钮。
- 显示了 “พนักงาน: test3 (officer3)” 和 “ยอดสุทธิที่ต้องจ่าย: 14,823.96 บาท”。
- 显示了 “โอนเข้าบัญชีธนาคาร” 部分，包含银行名称：ไทยพาณิชย์ และบัญชี：1234567890，以及 “ตัดออกเลขบัญชี” 按钮。
- 显示了 “ช่องทางการจ่ายเงิน” 部分，包含银行图标：K+ (กสิกรไทย), SCB (ไทยพาณิชย์), KTB (กรุงไทย), และ KB (กรุงเทพ)。
- 显示了 “ยืนยันจ่ายเงินแล้ว” 和 “ยกเลิก” 按钮。

หน้าทำเงินเดือน 2

- หน้าทำเงินเดือน 2 สร้างกลุ่มงวดเงินเดือน เพื่อใช้ในการแจ้งเตือนงวดเงินเดือนเมื่อถึงวันเวลา ที่ตั้งไว้
- หน้าทำเงินเดือน 3 เรียกดูงวดเงินเดือนที่ทำไว้ ที่เป็นสถานะรอจ่าย เพื่อจ่ายให้พนักงาน
- หน้าจ่ายงวดเงินพนักงาน หากเป็นการจ่ายผ่านบัญชีธนาคาร จะแสดง icon ธนาคารขึ้นมา หากจ่ายด้วยเงินสดจะไม่แสดง และจะบอกเป็นประเภทเงินสด
- กดยืนยันว่าจ่ายแล้วเพื่อเปลี่ยนสถานะงวดนั้น

หน้าทำเงินเดือน 3

หน้าจ่ายงวดเงินพนักงาน

3. ตั้งค่ากลุ่มงวดเงินเดือน

- กำหนดรอบแบบ monthly (วันจ่ายคงที่ เช่น 1/16/30/last) หรือ every_n_days (เริ่มจาก startDate เดินทุก N วัน)
- เลือก workdayOnly เพื่อเลี้ยงวันหยุด (สำหรับ monthly จะเลื่อนไปวันทำงานถัดไป)
- ใช้สำหรับแจ้งเตือนเมื่อถึงวันครบงวด หรือออกรายการทำเงินเดือนตาม “สมาชิกของกลุ่ม”

4. เรียกดูงวดที่ทำแล้ว & จ่ายเงิน

- เรียกดู payroll_runs / payroll_items ของงวดที่ผ่านมา
- อัปเดตสถานะการจ่ายรายคนและบันทึกบันทึกย่อ (note) หรือลับสถานะเป็น paid

5. รายงาน

- สรุปชั่วโมง/วันทำงาน, ชั่วโมงลา (แยกในโควตา/นอกโควตา), ยอดจ่ายสุทธิ ตามเดือนที่เลือก



คำสั่ง “รายงาน”

พนักงาน	ตำแหน่ง	ประเภทการจ่าย	อัตรา	วัน
test3	officer3	รายเดือน	15,000	
test1	officer1	รายวัน	80	
test2	officer2	รายวัน	300	

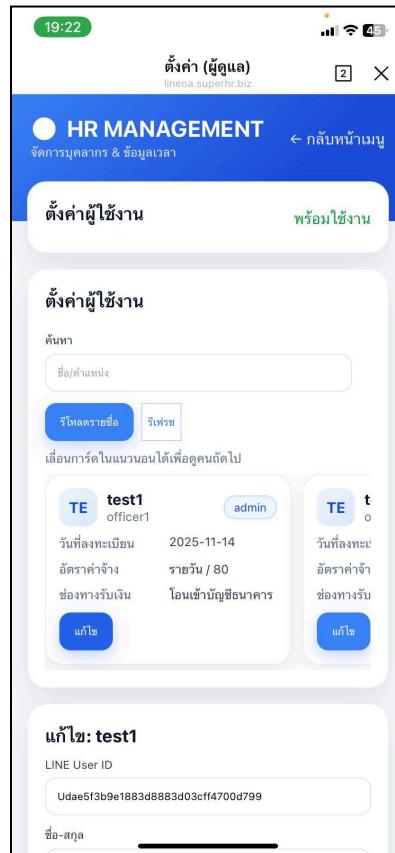
หน้าแสดงผลเฉียด และ
สรุปยอดในเดือนนี้ฯ
ตามงวดเงินเดือน

6. ตั้งค่าผู้ใช้งาน

- แก้ไขข้อมูลส่วนตัว, ค่าจ้าง, ชั่วโมงทำงานมาตรฐาน (คำนวณจาก shiftIn/shiftOut - breakMinutes)
- ตั้งค่าพักเบรก, โควต้าวันลา, lateGraceMin, ช่องทางจ่าย, allowances/deductions รายเดือน
- มอบหมายบทบาทเป็น admin



คำสั่ง “ตั้งค่า”



หน้าตั้งค่าผู้ใช้งาน

7. รีเซ็ตเมนู (กลับเป็น user) / เมนู admin

- คำสั่งสลับ Rich menu ของผู้ใช้เป็นเมนูพนักงาน หรือสลับกลับเป็นเมนูผู้ดูแล

8. ช่วยเหลือ admin

- แสดงคำสั่งทั้งหมดที่เกี่ยวกับการดูแลระบบและทำเงินเดือน

12) การกำหนดสิทธิ์ Admin ให้ผู้ใช้

ใช้เพื่อให้ผู้ใช้เข้าหน้าแอดมิน (เช่น Template Rich Menus) และอัปโหลดรูปได้

วิธีแนะนำ: ใช้สคริปต์ set-admin.js

- รันสคริปต์ tools/set-admin.js เพื่อเช็ค custom claims และอัปเดต users/{uid} (ตั้ง role + isAdmin)

วิธีสำรอง: แก้ isAdmin ตรง ๆ ในคอนโซล

ระบบ Frontend และกติกา Firestore/Storage จะอ้างอิงค่านี้ในการให้สิทธิ์

Task Assignment Bot

บอท “Task assignment” ทำงานบน LINE Messaging API โดยให้ผู้ใช้สั่งงาน/อัปเดตสถานะ/เพิ่มโน็ต/แก้เดดไลน์ ผ่านแชต LINE และเซิร์ฟเวอร์ Node.js จะประมวลผล (parse) คำสั่ง → เรียก Apps Script เป็น Data API เพื่ออ่าน/เขียนข้อมูลงานใน Google Sheet (ต่อ tenant) และส่งข้อความ/การ์ด Flex ตอบกลับ

Roles & Permissions

- **บทบาท:** developer, admin, supervisor, user
- **แก่ไขงานได้ถ้า:**
 - เป็น “ผู้สั่งงาน” (assigner) หรือ “ผู้รับงาน” (assignee) หรือ มีบทบาท developer/admin/supervisor
 - ตรวจสอบโดย canModifyTask(tenantRef, actorId, task)
- **ลิงก์ “จัดการผู้ใช้งาน”:** ถ้า role ไม่อยู่ในกลุ่มที่อนุญาต หรือ status ไม่ Active → บอทตอบปฏิเสธ (“ไม่ให้ลิงก์”)

Command Summary (สรุปคำสั่งทั้งหมด)

Registration

- คำสั่ง: ลงทะเบียน, สมัครใช้งาน, register
- ผลลัพธ์: เพิ่ม/อัปเดตผู้ใช้เป็น Active และแสดงเมนูเริ่มต้น

Assign Task (สั่งงาน)

- รูปแบบพิมพ์อิสระ:
 - @po ปรับรายงาน พรุ่งนี้ 09:00
 - @test ขอทำป้ายหน้าร้าน ก่อนบ่าย 3
 - @po ทำ rich menu วันนี้ ด่วน
- คีย์เวิร์ดเวลา: วันนี้, พรุ่งนี้, วันจันทร์/หน้า, เที่ยง/เที่ยงคึ่ง, บ่าย 3, 10/12 14:30, 10-12 9:00, ตัวเลขข้ามเดียวๆ
- ความเร่งด่วน: ด่วน/urgent ⇒ บันทึกโน๊ต [URGENT], ไม่รีบ/ค่อยทำ ⇒ บันทึกโน๊ต ไม่รีบ

Status Update (อัปเดตสถานะ)

- คำสั่ง:
 - done TASK_xxxx → สถานะ done
 - กำลังดำเนินการ TASK_xxxx → สถานะ doing

Deadline (กำหนดส่ง/แก้ไข)

- ตั้งครั้งแรก: ตั้งกำหนดส่ง TASK_xxxx: พรุ่งนี้ 15:00
- แก้เดดไลน์: แก้เดดไลน์ TASK_xxxx: ก่อนบ่าย 3

Detail / Note (รายละเอียด/โนํต)

- แก้รายละเอียด: แก้รายละเอียด TASK_xxxx: ปรับหัวข้อสรุปบทที่ 2
- เพิ่มโนํต: เพิ่มโนํต TASK_xxxx: เช็คกับลูกค้าก่อนส่ง

Reassign (โอนงาน)

- เปลี่ยนผู้รับ: เปลี่ยนผู้รับ TASK_xxxx: @newuser

Lists & Shortcuts (รายการงาน/เมนูลัด)

- ดูงานค้างทั้งหมด (ของฉัน): แสดงงาน pending/doing ของผู้ใช้
- งานของฉันวันนี้: งานถึงกำหนด “วันนี้” และยังไม่ done
- งานที่ฉันสั่ง: แสดงงานที่ผู้ใช้เป็นผู้สั่ง (assigner)
- ส่งงาน: ส่งคู่มือย่อ + รายชื่อผู้ใช้ที่ Active เพื่อช่วย mention

Users & Admin (ผู้ใช้/แอดมิน)

- ผู้ใช้งานทั้งหมด: แสดงรายชื่อแบบย่อ (เฉพาะ developer/admin/supervisor)
- จัดการผู้ใช้งาน: เปิดลิงก์หน้า Admin (จำกัดสิทธิ์; ถ้าเป็น user ระบบจะปฏิเสธและให้ติดต่อแอดมิน)
- ติดต่อแอดมิน: แสดงรายชื่อแอดมิน/ชุป และวิธีส่งข้อความ
 - รูปแบบ DM: dm @username ข้อความ หรือ ถึงแอดมิน @username ข้อความ

Misc (อื่นๆ)

- รีเซ็ตเมนู: รีเซ็ตเมนู, ตั้งเมนูแรก, รีเซ็ตเมนูของฉัน (เปลี่ยนเป็นเมนูเริ่มต้นของ OA)
- ปุ่มบันการ์ดงาน: “ เสร็จแล้ว”, “ กำลังทำ”, และ (ถ้าเปิดใช้) “เตือนงาน”

Architecture (โครงสร้างใน Project)

- **Client (LINE App):** ผู้ใช้กดเมนู (Quick replies) หรือพิมพ์ข้อความคำสั่งภาษาไทย
- **Server (Node.js / Express):** ไฟล์หลัก react-basic/server.js
 - รับ Webhook `/webhook/line` → `handleLineEvent(...)`
 - แยกโหมดข้อความ/เมนู และประมวลผลคอมมานด์เกี่ยวกับงาน
 - เรนเดอร์ Flex Messages (การ์ดงาน + ปุ่มสถานะ)
 - สร้าง Magic Link ไปหน้าเว็บแอดมิน (กรณี “จัดการผู้ใช้งาน”)
 - เรียก Apps Script ผ่าน `callAppsScriptForTenant(...)` เพื่อ CRUD งาน/ผู้ใช้
 - ตั้ง cron เตือนงานอัตโนมัติทุก 17:30 ว-ศ (Asia/Bangkok)
- **Parsers (NLP Lite):** react-basic/src/core/parsers.js
 - แยกชื่อผู้รับงาน (@username), รายละเอียด, เด็ดไลน์ภาษาไทย (“วันนี้/พรุ่งนี้/บ่าย 3/ก่อนบ่าย 3/เที่ยง/เที่ยงครึ่ง/วันจันทร์หน้า/10/12 14:30/10-12 9:00 ฯลฯ”), โน๊ต/ความเร่งด่วน (“ด่วน/ไม่รีบ”)
- **Data Layer (Google Apps Script):**
 - พิงก์ชัน: `list_users`, `get_user`, `upsert_user`, `list_tasks`, `get_task`, `upsert_task`, `update_task_status`, `add_note`, `tasks_of`, `ping`
 - เชิร์ฟเวอร์จะส่ง `sheet_id` ตาม tenant (ดึงจาก Firestore `tenants/{id}/integrations.taskbot`)

Key Files (สำหรับอ้างอิงใน project)

- Server: `react-basic/server.js`
 - Webhook: `app.post('/webhook/line', ...)` → เรียก `handleLineEvent(...)`
 - ฟังก์ชันสิทธิ์/อัปเดตงาน: `canModifyTask(...)`, `updateTaskFields(...)`, `getTaskById(...)`, `resolveAssignee(...)`, `renderTaskCard(...)`
 - เรียก Apps Script: `callAppsScriptForTenant(tenantRef, action, payload)`
 - Cron: `cron.schedule('30 17 * * 1-5', ...)` → `runDailyRemindersAllTenants()`
- Parsers: `react-basic/src/core/parsers.js`
 - `parseAssignLoose`, `parseAssign`, `parseStatus`, `parseSetDeadline`, `parseAddNote`, `parseReassign`, `parseEditDeadline`, `parseEditDetail`, `parseRemind`, `parseDeadline`, `parseNaturalDue`
- Admin UI (อ้างอิงสำหรับส่วนจัดการ):
 - `src/pages/AdminUsersSplitPage.jsx`
 - `src/pages/TaskAssignmentSettingsPage.jsx`
 - `src/api/client.js` (REST ผ่านเว็บแอป)

Data Flow

1. ผู้ใช้ส่งข้อความ/กดเมนูใน LINE → LINE POST มาที่ `/webhook/line`
2. `handleLineEvent(...)` ตรวจชนิด event + ดึง `tenantRef` ตาม destination → โอนด
ข้อความจะใช้ parsers และ intent
3. ถ้าเป็นคำสั่งเกี่ยวกับงาน → เรียก `callAppsScriptForTenant(...)` ไปที่ Apps Script
(อิง `APPS_SCRIPT_EXEC_URL` และ `APP_SHARED_KEY`)
4. ได้ผลลัพธ์กลับมา → ตอบผู้ใช้ด้วย **Flex Message** (การ์ดงาน + ปุ่มเปลี่ยนสถานะ/เตือน/ราย
ละเอียด) หรือข้อความธรรมดา
5. (กำหนดเวลา) cron 17:30 จ-ศ จะกราดทุก tenant ที่เปิดใช้ taskbot → ส่งสรุป/เตือนงานที่ “ถึง
กำหนดวันนี้และยังไม่เสร็จ” ให้ผู้เกี่ยวข้อง
6. ผู้ใช้ส่งข้อความ/กดเมนูใน LINE → LINE POST มาที่ `/webhook/line`
7. `handleLineEvent(...)` ตรวจชนิด event + ดึง `tenantRef` ตาม destination → โอนด
ข้อความจะใช้ parsers และ intent
8. ถ้าเป็นคำสั่งเกี่ยวกับงาน → เรียก `callAppsScriptForTenant(...)` ไปที่ Apps Script
(อิง `APPS_SCRIPT_EXEC_URL` และ `APP_SHARED_KEY`)
9. ได้ผลลัพธ์กลับมา → ตอบผู้ใช้ด้วย **Flex Message** (การ์ดงาน + ปุ่มเปลี่ยนสถานะ/เตือน/ราย
ละเอียด) หรือข้อความธรรมดา
10. (กำหนดเวลา) cron 17:30 จ-ศ จะกราดทุก tenant ที่เปิดใช้ taskbot → ส่งสรุป/เตือนงานที่ “ถึง
กำหนดวันนี้และยังไม่เสร็จ” ให้ผู้เกี่ยวข้อง

ภาคผนวก: วิธีตั้งค่าระบบเพื่อใช้งาน/ดิพลอยต่อ

1) สิ่งที่ต้องมี

- Firebase Project (Firestore + Storage + Authentication)
- LINE Developers:
 - LINE Login (Channel ID/Secret + callback URL)
 - Messaging API สำหรับเชื่อม OA
- โค้ดโปรเจกต์ (repo นี้) + Node.js LTS

ตำแหน่งไฟล์สำคัญในโปรเจกต์

- ค่าคอนฟิกฝั่งเว็บ: src.firebaseio.js
- ตัวแปรสภาพแวดล้อม: .env (มีตัวอย่างชื่อคีย์ให้พร้อม)
- Service Account (ใช้ฝั่ง server): firebase-admin.json (เก็บเป็น secret)

2) สร้าง/ตั้งค่า Firebase

1. เปิด Firestore (Native mode) + Storage
2. เปิด Authentication (อีเมล/LINE ตามที่ใช้)
3. สร้าง Service Account → ดาวน์โหลดไฟล์ JSON (ใช้เป็น secret)

Firestore rules

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {

    function authed() { return request.auth != null; }
    function isSelf(uid) { return authed() && request.auth.uid == uid; }

    // ใช้กับ doc ของ tenant เองเท่านั้น (resource == tenants/{tid})
    function isTenantMemberDoc() {
      return authed() &&
        resource.data.ownerUid == request.auth.uid ||
        (resource.data.members != null && request.auth.uid in resource.data.members)
    };

    // ใช้กับ subcollections ได้ tenant
    function isTenantMember(tid) {
      return authed() &&
        get(/databases/$(database)/documents/tenants/$(tid)).data.ownerUid == request.auth.uid ||
        request.auth.uid in get(/databases/$(database)/documents/tenants/$(tid)).data.members
    };

    function noSecretsInTenantDoc() {
      return !('channelSecret' in request.resource.data) &&
        !('accessToken' in request.resource.data) &&
        !('webhookSecret' in request.resource.data);
    }

    function onlyEditableTenantFieldsChanged() {
      return request.resource.data.diff(resource.data).changedKeys().hasOnly([
        'displayName','pictureUrl','chatMode','markAsReadMode','basicId','updatedAt'
      ]);
    }

    // ===== users =====
    match /users/{uid} {
      // อ่านได้ถ้าล็อกอิน (ใช้สำหรับค้นหาสมาชิก/แสดงโปรไฟล์)
      allow read: if authed();
      // สร้าง/แก้ไข ได้เฉพาะเจ้าของเอกสาร
    }
  }
}
```

```

        allow create, update: if isSelf(uid)
    allow delete: if false;
}

// ===== admin_templates =====
// ⌂ อันญาตให้ "อ่านได้ทุกคน" เพื่อให้ Guest เทืนหน้า Template
match /admin_templates/{tid} {
    allow read: if true;
    allow write: if authed() &&
        get(/databases/$(database)/documents/users/$(request.auth.uid)).data.isAdmin == true;
}

// ===== tenants =====
match /tenants/{tid} {
    // อ่านเฉพาะสมาชิกของ tenant เอง
    allow read: if isTenantMemberDoc();

    allow create: if authed()
        && request.resource.data.ownerUid == request.auth.uid
        && noSecretsInTenantDoc();
    allow update: if authed()
        && resource.data.ownerUid == request.auth.uid
        && noSecretsInTenantDoc()
        && onlyEditableTenantFieldsChanged();
    allow delete: if authed() && resource.data.ownerUid == request.auth.uid;

    // ห้าม client แตะ secrets
    match /secret/{doc} { allow read, write: if false; }

    // broadcasts
    match /broadcasts/{docId} {
        allow read: if isTenantMember(tid);
        allow create, update, delete: if authed() &&
            get(/databases/$(database)/documents/tenants/$(tid)).data.ownerUid == request.auth.uid;
    }
}

// richmenus (อ่านได้, เขียนผ่าน backend)
match /richmenus/{docId} {
    allow read: if isTenantMember(tid);
    allow create, update, delete: if false;
}

// live chat (อ่านได้เฉพาะสมาชิก; เขียนให้ backend ทำ)
match /liveSessions/{uid} {
    allow read: if isTenantMember(tid);
    allow create, update, delete: if false;
}
match /liveSessions/{uid}/messages/{mid} {
    allow read: if isTenantMember(tid);
    allow create, update, delete: if false;
}

// (อปปชั่น) เก็บ draft ของ Guest ใต้ /guests/{gid}/... ถ้าคุณใช้งาน

```

```
match /guests/{gid} {
  allow read, write: if true; // เก็บตัวยคุกที่ผ่าน server และ (ensureGuest) — ถ้าต้องการเข้มข้นให้ปรับเอง
  match /{coll}/{doc} {
    allow read, write: if true;
  }
}
```

Storage rules:

```
rules_version = '2';
service firebase.storage {
  match /b/{bucket}/o {

    function authed() { return request.auth != null; }
    function isImage() { return request.resource.contentType.matches('image/*'); }
    function small() { return request.resource.size < 6 * 1024 * 1024; }

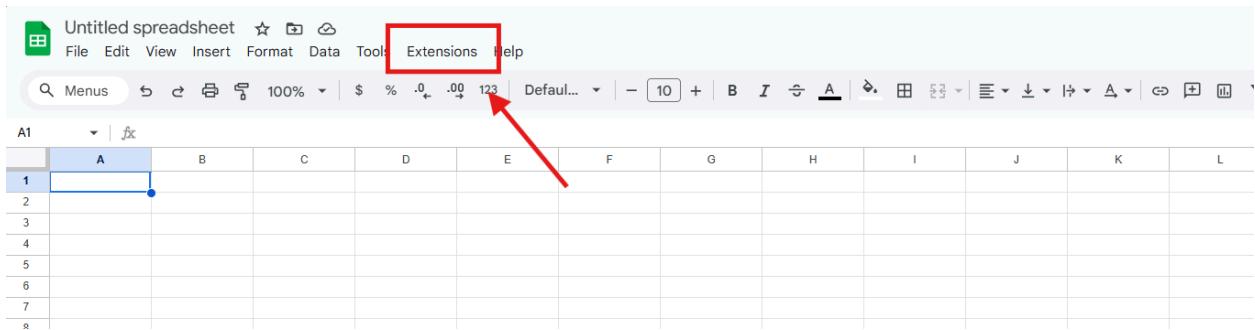
    match /public/admin-templates/{file=**} {
      allow read: if true;
      allow write: if authed() && isImage() && small();
    }
    match /admin/templates/{file=**} {
      allow read: if true;
      allow write: if authed() && isImage() && small();
    }

    match /imagemaps/{richId}/{file=**} {
      allow read: if true;
      allow write: if authed() && isImage() && small();
    }
    match /tenants/{tenantId}/imagemaps/{richId}/{file=**} {
      allow read: if true;
      allow write: if authed() && isImage() && small();
    }
    match /tenants/{tenantId}/rich-menus/{file=**} {
      allow read: if true;
      allow write: if authed() && isImage() && small();
    }

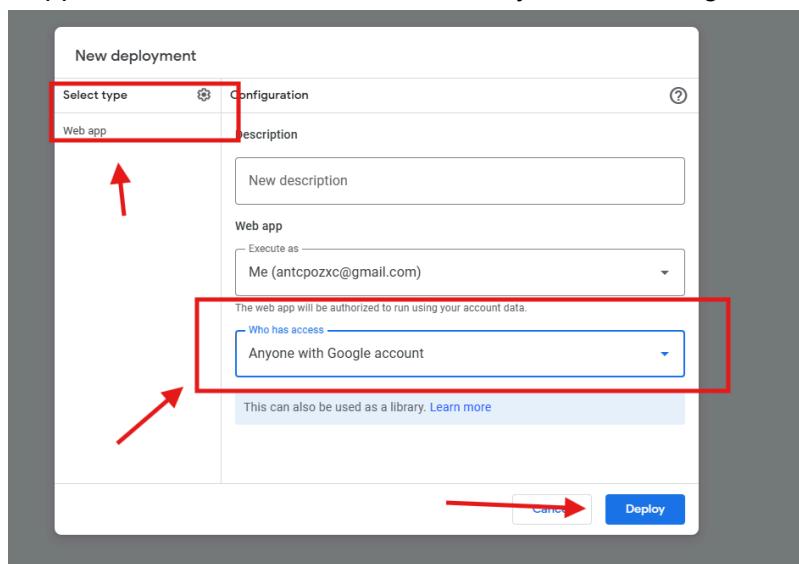
    match /tenants/{tenantId}/{rest=**} {
      allow read, write: if authed();
    }

    match /{allPaths=**} { allow read, write: if false; }
  }
}
```

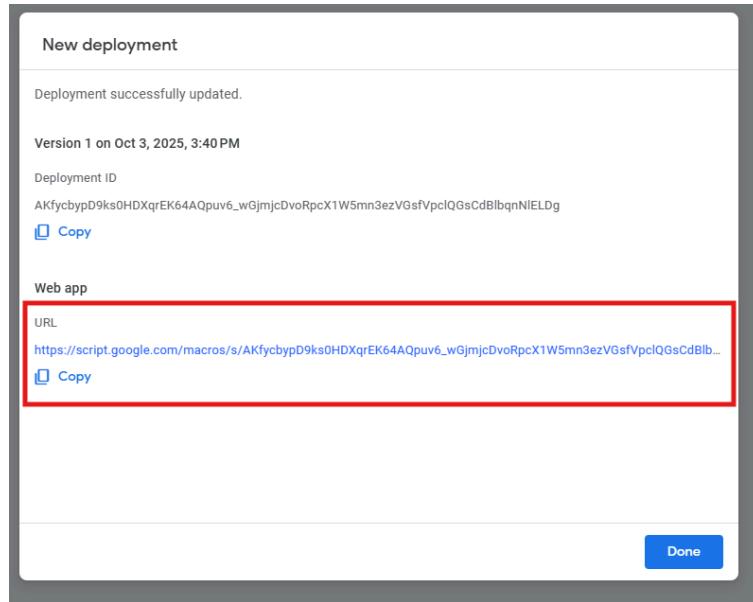
3) สร้าง/ตั้งค่า app-script เพื่อนำไปใช้งาน Task bot



1. สร้าง spreadsheet ใหม่
2. เลือกหัวข้อ Extensions -> App Script
3. ค้นหาไฟล์ ชื่อ “app-script” ใน Project ภายใต้ /react-basic
4. นำ code ชุดนี้มาวางใน [code.gs](#) และ New deploy
5. ตั้งค่าเป็น “Web app” และ Who has access ตั้งเป็น “ Anyone with Google account”



6. นำ link ที่ได้ลงท้ายด้วย /exec ไปใช้ตั้งค่าแทน env “APPS_SCRIPT_EXEC_URL”



4) ตั้งค่าผู้ใช้งาน (Client)

แก้ค่า Firebase config ใน `src/firebase.js` ให้เป็นของโปรเจกต์ใหม่ (ค่า apiKey/authDomain/projectId/appId/storageBucket) และ build ใช้งานได้เลย

ตัวอย่าง (ใส่ค่าจริงของโปรเจกต์ใหม่):

```
const firebaseConfig = {
  apiKey: '<WEB_API_KEY>',
  authDomain: '<PROJECT>.firebaseapp.com',
  projectId: '<PROJECT>',
  appId: '<APP_ID>',
  storageBucket: '<PROJECT>.storagebucket.app'
};
```

4) ตัวแปรสภาพแวดล้อม (Environment Variables)

ใช้ `.env` (สำหรับเครื่องนักพัฒนา) และ Environment ของโซสติง/Render (สำหรับโปรดักชัน) — โครงสร้างตัวอย่างมีในโปรเจกต์แล้ว

KEY	คำอธิบาย
PUBLIC_APP_URL	URL ของเว็บ (เช่น <a href="https://<your-app>.onrender.com">https://<your-app>.onrender.com)
LINE_LOGIN_CHANNEL_ID	จาก LINE Login
LINE_LOGIN_CHANNEL_SECRET	จาก LINE Login
LINE_LOGIN_CALLBACK_URL	<code><PUBLIC_APP_URL>/auth/line/callback</code>
GUEST_JWT_SECRET	secret สำหรับ guest token
CRON_KEY	คีย์เรียก endpoint งาน cron ภายในระบบ (ถ้ามี)
DEBUG_WEBHOOK	1 เพื่อเปิด 0 ออก (อปชัน)

เลือกหนึ่งวิธีให้ Service Account

`FIREBASE_SERVICE_ACCOUNT_JSON`

(แนะนำบน Render) วางค่า JSON ที่เก็บของ Service Account ลงในตัวแปรนี้

`GOOGLE_APPLICATION_CREDENTIALS`

(สำหรับเครื่อง dev) path ไฟล์ service account ในเครื่อง เช่น `C:/path.firebaseio-admin.json`

5) ตั้งผู้ดูแลระบบเริ่มต้น (Custom Claims)

ให้มืออย่างน้อย 1 คนเป็น **developer** เพื่อจัดการสิทธิ์ในหน้า *Administrator management* วิธีตั้ง:

```
# ชี้ service account ก่อน (ถ้าทำงานเครื่อง)
export GOOGLE_APPLICATION_CREDENTIALS="C:/path.firebaseio-admin.json"
```

```
# แก้ UID ใน tools/set-admin.js และรัน
node tools/set-admin.js
# จะตั้ง role = developer ให้กับ UID ที่ระบุ
```

6) Deploy

Local dev

```
npm i
# สร้าง .env จากตัวอย่าง และใส่ค่า
npm start    # frontend
node server.js # backend (ถ้าแยก)
```

7) เช็กลิสต์ทดสอบหลังเซ็ตอัป

- ล็อกอิน → เห็นเมนูตามบทบาท

- ตั้ง Developer สำเร็จ → หน้า **Admin** → **Administrator management** ใช้งานได้ (เปลี่ยนบทบาท/ลบผู้ใช้ตามสิทธิ์)
- เชื่อม OA ได้จากหน้า **Accounts**
- หน้า **Rich Menu**: เปิด **Design Image (Layers)** → export → save ได้
- อัปโหลดไฟล์ขึ้น Storage ได้ตาม rules
- (ถ้าใช้ LINE Login) ล็อกอินผ่าน LINE ได้, callback URL ลูกต้อง
- ทดสอบการทำงาน Task Assignment bot

หมายเหตุสำหรับผู้รับช่วงงาน

- รายการข้อคีย์ .env ตรงกับที่โปรเจกต์ใช้งานจริงแล้ว (ดูไฟล์ .env ตัวอย่างใน repo เพื่อ pattern ของค่าที่ต้องใส่)
- ไม่ต้องนำไฟล์คีย์ริงใส่ repo—ให้ใส่ใน Environment/Secrets ของระบบเท่านั้น (ตำแหน่งไฟล์ service account ที่ใช้ผู้เชื่อมต่อ server ดูตัวอย่างได้ในโปรเจกต์)

Environment/Secrets ที่ใช้งานจริงในระบบ

KEY	VALUE
APP_JWT_SECRET	change_me_strong_secret
APPS_SCRIPT_EXEC_URL	https://script.google.com/macros/s/AKfycbxWPsBpgPkpK_jZvrcwmn4mtb0gnA09hfasxhswoC01WtH6xP0cPyMjwnpkWrUCg/exec
APPS_SCRIPT_SHARED_KEY	oifjhweoirjgfowrejgowerngojwpasdasdwe
CRON_KEY	mysupersecretkey_2025_XYZ123
DEBUG_WEBHOOK	1
FIREBASE_SERVICE_ACCOUNT_JSON	{"type": "service_account", "project_id": "line-rich-menus-web", "private_key_id": "fcfb5a6c3125fb167788a670e136900de0ab8a", "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvgADANBgkqhkiG9w0BAQEFAASBgmg9SkaGEAcIBAQDQjD02WYRPoRe1z2pQGadZwdbd0w74Ybse6okpL7LzC2Ra8U9sch2AwGeL3LF+NT1XmEY5dg\nqWPn26MNKL0g7m7zK1P68oy3awYygb/GXEGSS30NdvfsJoh0Kz2p8zX28M5LwYy1NTK1nv0c60JUv2k1Mpnhoxed0e0cy1Ay7xfz8gOpPEPxAS3B0Rx14t7tFnH\nPgPKMlVNxvDRn0cyl12P4z2sWeCNr05jKK0d04UxRp2tawPLzLk57609+jKyW006nsmplHz2+wSzqIH/DzTzRwvCAge03CEfx5jEZw6/hj2ewtmHs+rVNa\nv1qK+mwd1nwWtln1u1gPAEAoggAEAB31YtrR6G70K6nBAS1+fw3LGS9My6Dw0dr1Um1E9vbx/nMs2h07UVQP+5ELR4+ecBFz2lpxrqxa7wLbo/Vo7z1b03RddRxz\n+F3Z0SLhp9xUnpEcveyW020my5GSAFhUEduA5x1o0eJaKbxBvREHUJPjJEV3HEinPEX3xKaGKnMAlPs/0x3/XvqudY14DipzBgt1BL0uFdfn40NFzF0KKVm\nWxV4JHjNtR0DHwvnSM6QQLxU0VECMo0j1011as82NIRKAUL957D7XUfx/XUm1Ts1adkF1nHmBjGj1nHs383t99Pfix1JnHYSVBExv418aQGtf0fs2wxEkQK8gQ\nDRqCMj0mql0Q2jPzK2lcnol2y6gGuyLVIUKQK5gysal0cBAw2qJUL2cA2Z0kIa1aiJ30U1cc0g9Fy0rncW00/n8vEWi+CBPw2UvBqNpQ2ksrNbtkLhpPBugdH2zmk\nPx/RHdmrjz01SNBt4d2Pq1n/vp708pxMdn2cPHC0Ykd8PPQKbgD0ABHEivVv-L0p0yAemb/qk+AjDVi1wHUVv/nG0VpHloytoXLmfNzulNKRXmEvMfpzKm28MX\nctQPr5iuhk2Drd0M14ezCeK1n+kPzgeyAIDYijfygkfcu2xcgt/byk841pu10msttWfRgjsW4e2s9bQk1n1u1ZaxpA120Kbgf1RcyK484dqhRqRgPluybSxGt\nDNKUoxzD1eb1295kdNvE64BkuBu/nAyb7Fx9sRyHmlV0u0Lm919xv9u1taDqQkTwJ9xxu0wDwjlLxMwHe134nDvtaa2pr+b0cooE8osnL17fjCmzSSw9\nitrgao91Mbj6wCRimk5AoGBAlun/n207dE6uiuEvDbZPG/Vfd0QG/CcekgqnTfIK/3DkjcqbnsohuPqtyxxv99/nFa77XARCcKgn296gbtX0Xz/dMGoEvn\nojwInqdJlMuYemmlSoaq9Z/37-dvBB3znn-nL-Vj/5Y0nnoC7Q0UEin5j5u4zybeaN07uBaGBAL0J4oASGeoS0Pyamq3nqf1qfChNpG9/14Cix06s1rvEbh\nik2dry07UIABhukMn0h4uHoz11Qu1Qd8ttnh9JmzE63U1xitor563L0GMyvk634Yug27vs5dnhdBxE2KbHMnDrqexJttPNng0wANwml1TrwIe50036xB1c0\n-----END PRIVATE KEY-----", "client_email": "firebase-adminsdk-fbsvc@line-rich-menus-web.iam.gserviceaccount.com", "client_id": "115583372207200077279", "auth_url": "https://accounts.google.com/o/oauth2/auth", "token_url": "https://oauth2.googleapis.com/token", "auth_provider_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509.firebaseio-admins-fbsvc@line-rich-menus-web.iam.gserviceaccount.com", "universe_domain": "googleapis.com"}

GUEST_JWT_SECRET	aEk0HiiZ7Zak0FdFYcRmjWg5ReQ82MSxPUzByCvtW7VI_VmVxhakZ7ZXtoTN161aKG_MH096PydDpk_de70g
LINE_LOGIN_CALLBACK_URL	https://line-rich-menu-web.onrender.com/auth/line/callback
LINE_LOGIN_CHANNEL_ID	2007922762
LINE_LOGIN_CHANNEL_SECRET	844c5e01afc8d95d6dedda84aa0c0d5
MAGIC_TTL	24h
PUBLIC_APP_URL	https://line-rich-menu-web.onrender.com
SHEETS_SHARED_SECRET	my_shared_token_9c0f1a

- APP_JWT_SECRET – กรุณาเข้ารหัส JWT ภายในแอป (เช่น magic link)

- APPS_SCRIPT_EXEC_URL – URL ของ Google Apps Script (doPost “exec”) ที่ผู้ใช้ฟีเวอร์เรียกใช้งานเป็น Data API
- APPS_SCRIPT_SHARED_KEY – โทเคนที่ทั้ง Server และ Apps Script ใช้ร่วมกันเพื่อตรวจสอบสิทธิ์
- CRON_KEY – คุณจะเรียก route งานตามเวลา (cron) กับบุคลาภายนอกยิง
- DEBUG_WEBHOOK – เปิด/ปิด log โหมดตีบักของ LINE webhook (1 = เปิด)
- FIREBASE_SERVICE_ACCOUNT_JSON – JSON Service Account ของ Firebase Admin SDK (ใช้ verify/issue token ฯลฯ)
- GUEST_JWT_SECRET – คุณจะออก/ตรวจสอบ JWT สำหรับโหมด guest (ถ้าใช้)
- LINE_LOGIN_CHANNEL_ID/SECRET/CALLBACK_URL – ค่าจาก LINE Login (OIDC) สำหรับเข้าสู่ระบบผ่าน LINE
- MAGIC_TTL – อายุของ magic link (เช่น 24h, 2h)
- PUBLIC_APP_URL – base URL แบบ public ของเว็บ/เซิร์ฟเวอร์ (Render/ngrok) ใช้ประกอบลิงก์ต่างๆ
- SHEETS_SHARED_SECRET – โทเคนผึ่งชีท/สคริปต์ (เทียบเท่า shared key; ดูหมายเหตุด้านล่าง)

รายละเอียดที่ลະດັບ

APP_JWT_SECRET

- ใช้ท่าอะไร: เข้ารหัส/คอดรหัส JWT ภายใน (เช่น โทเคนของ magic link /auth/magic)
- รูปแบบ: สตริงยาวสัมแข็งแรง
- ข้อควรระวัง: เปลี่ยนค่าทันทีถ้าส่งสัญญาให้หละ; เปลี่ยนแล้วโทเคนเดิมจะใช้ไม่ได้

APPS_SCRIPT_EXEC_URL

- ใช้ท่าอะไร: ชี้ไปยัง Apps Script Web App รุ่นที่ deploy แล้ว (exec) ที่รับ POST จากเซิร์ฟเวอร์
- หาได้ที่: Apps Script → Deploy → “Web app” → URL ที่ลงท้ายด้วย /exec
- ข้อผิดพลาดพบบ่อย: ใช้ URL ผิด (เช่น dev “/dev” แทน “/exec”) หรือยังไม่ได้ Deploy

APPS_SCRIPT_SHARED_KEY

- ใช้ท่าอะไร: เป็น shared secret ระหว่าง Server ↔ Apps Script เพื่อตรวจสอบคำสั่ง (เช่น upsert_user, upsert_task)
- การตั้งค่า: ต้องใส่ค่า เดียวกัน ทั้งฝั่ง Server (env นี้) และฝั่ง Apps Script (เช่น Script Properties)
- หมายเหตุ: ในโค้ดบางที่อาจใช้ชื่อ APP_SHARED_KEY แทน—ให้ตรวจสอบว่าฟังก์ชันอ่านจากตัวไหน ถ้าไม่แน่ใจ ให้ตั้ง ทั้งสอง เป็นค่าเดียวกัน

CRON_KEY

- ใช้ท่าอะไร: ใส่เป็นพารามิเตอร์/เขตเดอร์เวลาเรียก endpoint cron ภายใน (เช่น summary ตอน 17:30) เพื่อกันคนนอกยิง
- คำแนะนำ: สุ่มยาวๆ และหมุนเวียนเมื่อมีการแซร์ฟิร์ระบบอื่น

DEBUG_WEBHOOK

- ใช้ท่าอะไร: 1 = เปิด log เพิ่มสำหรับ LINE webhook (ช่วยดีบัก)
- คำแนะนำ: เปิดเฉพาะช่วงดีบัก/สเตจ; โปรดปิดในโปรดักชันเพื่อลด noise

FIREBASE_SERVICE_ACCOUNT_JSON

- ใช้ท่าอะไร: ใส่ JSON service account แบบเดิมสำหรับ firebase-admin
- รูปแบบ: ใส่ JSON ทั้งก้อน (มี private_key ฯลฯ) ในค่า env เดียว; ใน Render สามารถ JSON ได้ตรงๆ (ไม่ต้อง escape \n)
- คำเตือน: ความลับสูงสุด—ห้ามแชร์/คอมมิตลง repo; รู้ว่าหลังต้อง re-key/หมุนกุญแจด้วย

GUEST_JWT_SECRET

- ใช้ท่าอะไร: กรณีมี flow “guest” (เช่น หน้าอ่าน-only) ใช้เข้ารหัส/กودรหัส JWT ของโหมดนั้นแยกจาก APP_JWT_SECRET
- ถ้าไม่ใช้: ตั้งค่าไว้ก็ไม่กระทบ แต่ควรเก็บเป็นความลับ เช่นกัน

LINE_LOGIN_CHANNEL_ID / LINE_LOGIN_CHANNEL_SECRET /

LINE_LOGIN_CALLBACK_URL

- ใช้ท่าอะไร: สำหรับ LINE Login (OIDC) ผ่านเว็บ (เช่น /auth/line/callback) เพื่อแลก access_token และอ่าน userinfo
- ต้องตรงกัน: ค่า CALLBACK_URL ต้องตรงกับที่ตั้งใน LINE Developers (Channel → LINE Login → Callback URL)
- อาการผิดพลาด: callback 400/401 หรือแลก token ไม่ได้ → มักเกิดจาก secret>ID/URL ไม่ตรง

MAGIC_TTL

- ใช้ท่าอะไร: กำหนดอายุของ magic link ที่ออกจาก /auth/magic เช่น 2h, 24h
- ข้อควรระวัง: นานเกินไปเสี่ยง; สั้นเกินไปผู้ใช้จะหมดอายุเร็ว เลือกตาม use case

PUBLIC_APP_URL

- ใช้ท่าอะไร: base URL ของระบบที่ ผู้ใช้เข้าถึงได้จริง (เช่น Render) ใช้ประกอบ redirect / สร้างลิงก์ / ตรวจที่มา
- ตัวอย่าง: <https://line-rich-menu-web.onrender.com>
- ข้อผิดพลาดพบบ่อย: ลืมอัปเดตจาก ngrok ชุดเก่า → ลิงก์/redirect พัง

SHEETS_SHARED_SECRET

- ใช้ท่าอะไร: โทเคนผึ่งชีท/สคริปต์ (บางโปรเจกต์ให้ข้อตัวนี้เทียบกับ APPS_SCRIPT_SHARED_KEY)
- ข้อควรระวัง: ให้ตั้ง ค่าเดียวกัน กับ APPS_SCRIPT_SHARED_KEY เพื่อไม่สับสน หรือเลือกใช้เพียงตัวเดียวให้ตรงกับโค้ดจริง

Sanity checks (หลังตั้งค่า)

PUBLIC_APP_URL ถูกต้อง

- ลองเปิดหน้าเว็บหลัก และ endpoint ที่เกี่ยวข้อง (เช่น /healthz ถ้ามี)

LINE Login

- กด Sign in → callback ต้องทำงาน, ไม่มี 400/401
- ตรวจสอบ log: ได้ code, แลก token สำเร็จ, อ่าน userinfo ได้

Apps Script API

- เรียกคำสั่งง่ายๆ (เช่น ping/list_users) จากแอป → ถ้า 403 ให้ตรวจสอบ APPS_SCRIPT_SHARED_KEY และ Script Properties ผ่าน GAS

Cron

- เรียก endpoint cron พร้อม CRON_KEY → ได้ 200 และบันทึก log

Rich menu / Messaging

- ส่งข้อความทดสอบใน OA → webhook log แสดง event และตอบกลับได้

ตัวอย่าง .env

```
#env

# App
PORT=3000
PUBLIC_APP_URL=https://honeydewed-dolessome-camilla.ngrok-free.dev
# https://line-rich-menu-web.onrender.com

DEBUG_WEBHOOK=1
# LINE Login (จาก LINE Developers -> LINE Login channel)
LINE_LOGIN_CHANNEL_ID=2007922762
LINE_LOGIN_CHANNEL_SECRET=844c5e01afc8d95d6dedda64aaf0c0d5
# ต้องตรงกับที่ตั้งใน LINE Developers
LINE_LOGIN_CALLBACK_URL=https://honeydewed-dolessome-camilla.ngrok-free.dev/auth/line/callback
# https://line-rich-menu-web.onrender.com/auth/line/callback

GUEST_JWT_SECRET=aEkC6HI1ZZWak0oFdFYcRmjMG5ReQ82MSxPUr8yCVtW7VI_VmVx
hkaZZ7ZXoTNI6iaKG_MHO96PydDpk_de7Dg

# ตัวอย่างทดสอบล็อกอิน: https://<ngrok-id>.ngrok.io/auth/line/callback

# Firebase Admin
FIREBASE_PROJECT_ID=line-rich-menus-web

CRON_KEY=mysupersecretkey_2025_xYZ123

# เลือกหนึ่งวิธีในการให้ Service Account แก่ Firebase Admin

# Option A: ระบุ path ไฟล์ service account บนเครื่อง/เซิร์ฟเวอร์
# GOOGLE_APPLICATION_CREDENTIALS=/absolute/path/to/service-account.json

# Option B: วาง JSON แบบ base64 (แนะนำกับ Render/Hosting)
GOOGLE_APPLICATION_CREDENTIALS=C:/myWeb/secrets.firebaseio-admin.json

#
APPS_SCRIPT_WEBAPP_URL=https://script.google.com/macros/s/AKfycbxYWsPsBpgPpKP_j
Zvrcwmn4mtbOgnAO9hfaxhswo8COIWt5hG6xPOcPyMJwnpkWrUCg/exec

# SHEETS_SHARED_SECRET=my_shared_token_9c0f1a
```

```
# Apps Script WebApp (Google Sheets API ที่คุณทำไว้)
#
APPS_SCRIPT_EXEC_URL=https://script.google.com/macros/s/AKfycbxYWPsBpgPpKP_jZvr
cwmn4mtbOgnAO9hfaxhswo8COIWt5hG6xPOcPyMJwnpkWrUCg/exec
APP_JWT_SECRET=change_me_strong_secret
# APP_SHARED_KEY=oifjhweorijgfowrejgowerngojwpasdasdwe

# ใช้แค่ 2 ตัวนี้สำหรับ Apps Script
# (Task assignment)
APPS_SCRIPT_EXEC_URL=https://script.google.com/macros/s/AKfycbxYWPsBpgPpKP_jZvr
cwmn4mtbOgnAO9hfaxhswo8COIWt5hG6xPOcPyMJwnpkWrUCg/exec
APPS_SCRIPT_SHARED_KEY=oifjhweorijgfowrejgowerngojwpasdasdwe

# (Time Attendance)
# ----- Time Attendance (ใหม่) -----
APPS_SCRIPT_EXEC_URL_TA=https://script.google.com/macros/s/AKfycbwzNH3Z4Nfddxuzq
yhr9vkBUYPhoZBNj3_CaSmqXthSwIqTFG_N91i78_i9JfwerlYw/exec
APPS_SCRIPT_SHARED_KEY_TA=oifjhweorijgfowrejgowerngojwpasdasdwe
```

MAGIC_TTL=24h

IAPP_API_KEY=zK4YfOctouSzqn3ZLqGUscUOdt19WMAV

LIFF_TA_CLOCK_ID=2007922762-89eg5nZz
LIFF_TA_LEAVE_ID=2007922762-1nLrY2xe
LIFF_TA_ADMIN_ID=2007922762-kPXrR0K4

THAI_FONT_PATH=./assets/fonts/NotoSansThai-Regular.ttf
THAI_FONT_BOLD_PATH=./assets/fonts/NotoSansThai-Bold.ttf

```
# ให้ job นี้คุณเวลาเอง แยกจากงานเดิม
TA_REMIND_ENABLED=true
TA_REMIND_CRON=0 9 * * 1-5
# (ถ้าเซิร์ฟเวอร์รันบน UTC)
TZ=Asia/Bangkok
```

GAS (file Google App Script)

Task Assignment bot :

Time Attendance :