เอกสารขอบเขตงาน (Terms of Reference) v1.0

โครงการ: The D(r)ink District – แพลตฟอร์มระบบจองสนามกีฬาแบบหลายสาขา & กระเป๋าเงินดิจิทัล
วันที่: 25 สิงหาคม 2025

0. บทสรุปผู้บริหาร

โครงการนี้เป็นการพัฒนาเว็บแอปพลิเคชัน PWA (Progressive Web App) ที่ทำงานผ่าน LINE LIFF ซึ่งออกแบบมาสำหรับการ **จองสนามกีฬา** (เช่น สนามปิกเกิลบอล) ในหลายสถานที่ (หลายสาขา) ควบคู่ไปกับระบบ **ขายสินค้า/บริการเสริม** (อาหาร เครื่องดื่ม Merchandise แพ็กเกจโปรโมชั่น) และระบบ **กระเป๋าเงินดิจิทัล** สำหรับชำระเงินภายในแพลตฟอร์มเอง โดยทั้งหมดจะผสานการทำงานอยู่ภายใต้ LINE Official Account (LINE OA) ของโครงการ

เฟส 1 ของระบบจะครอบคลุมฟังก์ชันหลักดังต่อไปนี้:

- **ระบบจองสนาม** หลายสถานที่ พร้อมการสร้าง PIN เข้าใช้สนาม (รองรับการแชร์ให้ผู้ร่วมจองผ่าน LINE อย่างปลอดภัย)
- **ระบบชำระเงินผ่านกระเป๋าเงินดิจิทัล** (District Credits) ที่ผู้ใช้สามารถเติมเงินเข้าไปได้ โดยรองรับการสแกนเพื่อจ่ายเงิน (wallet fast track) ผ่าน LINE
- **ระบบบริหารจัดการหลังบ้าน (Admin Dashboard)** สำหรับกำหนดกฎการใช้งานต่างๆ เช่น เงื่อนไขการชำระเงิน การตั้งค่าแพ็กเกจโปรโมชั่น การจัดการข้อมูลสนาม/สินค้า ตลอดจนดูรายงานการขายและข้อมูลทางบัญชี
- **การเชื่อมต่อกับระบบบัญชี** (ผ่าน API ของ FlowAccount) เพื่อบันทึกธุรกรรมทางการเงิน (เช่น ยอดเงินในกระเป๋าเงินที่เติม/ใช้) เข้าสู่ระบบบัญชีของบริษัทโดยอัตโนมัติ
- **การเชื่อมต่ออุปกรณ์ประตูอัจฉริยะ** (Igloohome Deadbolt 2S) โดยระบบสามารถสร้างและลบรหัส PIN สำหรับกุญแจดิจิทัลได้แบบอัตโนมัติ และแจ้ง PIN นั้นให้ผู้จองทราบผาน LINE ในช่วงเวลาที่จองไว้เท่านั้น
- **ระบบแจ้งเตือนผ่าน LINE OA** เพื่อส่งข้อมูลสำคัญถึงผู้ใช้ เช่น การยืนยันการจอง รหัสเข้าพื้นที่ (PIN) ยืนยันการชำระเงิน สถานะคำสั่งซื้อ และแจ้งเตือนเครดิตใกล้หมดอายุ

แพลตฟอร์มนี้ถูกออกแบบให้ **เน้นการใช้งานบนมือถือเป็นหลัก** (Mobile-first) ผ่านแอป LINE โดยกลุ่มเป้าหมายหลักคือชาวต่างชาติที่อาศัยในประเทศไทย (Expat) ซึ่งถนัดการใช้งานภาษาอังกฤษ ดังนั้นระบบจะใช้ภาษาอังกฤษเป็นค่าเริ่มต้น แต่ก็มีการรองรับภาษาไทยเพื่อผู้ใช้คนไทยด้วยเช่นกัน

> *หมายเหตุ:* โครงการมีการวางแผนฟังก์ชันเพิ่มเติมใน **เฟส 2** อันได้แก่ ระบบวิเคราะห์ข้อมูล (Analytics), ระบบสมาชิกและสะสมแต้ม (Membership/Loyalty), โปรแกรมแนะนำเพื่อน (Referral Program), การเชื่อมต่อกับระบบจัดอันดับผู้เล่น (DUPR Rating) และการอัปเกรดระบบประตูเป็นการสแกน QR Code เป็นต้น อย่างไรก็ตาม ฟังก์ชันเหล่านี้อยู่นอกขอบเขตของการพัฒนาในครั้งนี้ แต่ควรถูกคำนึงถึงในการออกแบบระบบเพื่อรองรับการเพิ่มในอนาคต

1. ขอบเขตของโครงการ

ขอบเขตการพัฒนา (Phase 1): ทีมพัฒนาจะต้องพัฒนาระบบตามฟังก์ชันต่างๆ ที่ระบุในเอกสารฉบับนี้ ซึ่งครอบคลุมทุกโมดูลหลัก 7 ส่วน ได้แก่ ระบบจองสนาม, ระบบแพ็กเกจและบริการเสริม, ระบบโปรโมชั่น, ระบบกระเป๋าเงิน & ชำระเงิน, ระบบหลังบ้าน, ระบบแจ้งเตือน, และระบบความปลอดภัย/บันทึกการใช้งาน (รายละเอียดในหัวข้อ 7). ทุกฟังก์ชันจะต้องทำงานบน LINE LIFF PWA และเชื่อมต่อกันเป็นแพลตฟอร์มเดียวอย่างสมบูรณ์

งานพัฒนานี้ไม่รวมฟีเจอร์ในเฟสถัดไปที่ได้กล่าวถึงในบทสรุป (เช่น Loyalty, Membership, Referral, Analytics, DUPR) แต่ระบบที่พัฒนาจะต้องถูกออกแบบมาให้สามารถ **ขยายเพิ่มเติมในอนาคต** เพื่อรองรับฟีเจอร์เหล่านั้นได้โดยไม่ต้องรื้อทำใหม่

สิ่งที่ไม่อยู่ในขอบเขต: การจัดหาฮาร์ดแวร์เพิ่มเติม (เช่น อุปกรณ์ POS จริง เครื่องสแกน หรืออุปกรณ์ประตู) ไม่รวมอยู่ในงานครั้งนี้ และค่าใช้จ่ายในการจัดหาอุปกรณ์เหล่านั้นอยู่นอกงบประมาณที่กำหนด

2. ระยะเวลาในการพัฒนาและการสนับสนุน

- **ระยะเวลาพัฒนา:** ไม่เกิน **40 วันปฏิทิน** นับจากวันเริ่มงาน (รวมวันหยุดราชการและวันหยุดสุดสัปดาห์) สำหรับการพัฒนาระบบทั้งหมดจนครบฟังก์ชันตามขอบเขตที่กำหนด
- **ระยะเวลาทดสอบ:** หลังสิ้นสุดการพัฒนา ทีมพัฒนาจะต้องดำเนินการทดสอบระบบร่วมกับผู้ว่าจ้างเป็นเวลา **5 วัน** เพื่อตรวจสอบการทำงานของระบบ (User Acceptance Test) และแก้ไขข้อบกพร่องต่างๆ ให้ระบบพร้อมใช้งานจริงอย่างสมบูรณ์
- **การสนับสนุนหลังส่งมอบ:** หลังจากส่งมอบงานและผ่านการทดสอบเรียบร้อยแล้ว ให้มีช่วง **ระยะประกัน/สนับสนุน 60 วัน** ซึ่งในช่วงนี้หากพบปัญหาในการใช้งานหรือต้องการความช่วยเหลือแก้ไขใดๆ ทีมพัฒนาจะต้องเข้ามาช่วยเหลือตามคำร้องขอ โดยการสนับสนุน/แก้ไขงานในช่วงนี้จะมี **ค่าตอบแทนเพิ่มเติมเป็นกรณีตามแต่ละครั้ง** ที่เข้าดำเนินการ (ตามเงื่อนไขที่จะตกลงร่วมกัน)

3. งบประมาณ

งบประมาณสำหรับโครงการนี้กำหนดไว้ที่ **200,000 บาท** (สองแสนห้าหมื่นบาทถ้วน) ซึ่งครอบคลุมค่าใช้จ่ายในการพัฒนาซอฟต์แวร์ทั้งหมด รวมถึงการออกแบบระบบ การทดสอบ และการปรับแก้ตามขอบเขตงานที่ระบุ **โดยไม่รวม** ค่าใช้จ่ายในการจัดซื้ออุปกรณ์หรือฮาร์ดแวร์เพิ่มเติมใดๆ เช่น เครื่อง POS แท็บเล็ต เครื่องสแกน หรืออุปกรณ์ IoT อื่นๆ ที่อาจต้องใช้ในการใช้งานจริง (หากมีการจัดซื้อจะเป็นงบเพิ่มเติมนอกเหนือจาก 200,000 บาท)

4. ข้อกำหนดทางเทคนิคและการติดตั้งระบบ

- ระบบทั้งหมดจะต้องทำงานผ่าน **LINE Official Account (LINE OA)** ของโครงการ โดยใช้เฉพาะ **LINE LIFF** (LINE Front-end Framework) ในการแสดงผล PWA และใช้ **LINE Login** ในการยืนยันตัวตนผู้ใช้ (ไม่ต้องพัฒนาระบบล็อกอินแยกต่างหากหรือรองรับการล็อกอินรูปแบบอื่น)
- ระบบจะต้องพัฒนาในรูปแบบ **Progressive Web App (PWA)** เพื่อให้ผู้ใช้สามารถเข้าผ่าน LINE ได้ทันทีโดยไม่ต้องติดตั้งแอปพลิเคชันเพิ่มเติม และควรรองรับการใช้งานบนมือถือเป็นหลัก (responsive กับขนาดหน้าจอมือถือและแท็บเล็ต)

- **การรองรับหลายภาษา:** ระบบต้องรองรับการแสดงผล 2 ภาษา ได้แก่ **ภาษาอังกฤษ** (เป็นค่าเริ่มต้นที่ผู้ใช้จะเห็นเมื่อเข้าใช้งานครั้งแรก) และ **ภาษาไทย** ผู้ใช้ควรสามารถสลับภาษาได้ตามต้องการ โดยเนื้อหา ข้อความ และรูปภาพประกอบต่างๆ ควรเตรียมสำหรับทั้งสองภาษา
- **การติดตั้งและโฮสติ้ง:** ทีมพัฒนาจะต้องรับผิดชอบในการ **Deploy** ระบบขึ้นเซิร์ฟเวอร์และจัดการ **Hosting** แบบครบวงจร จัดเตรียมและตั้งค่า **โดเมนเนม** ที่จะใช้เข้าถึงระบบ ตลอดจนโครงสร้างพื้นฐานเบื้องต้นที่จำเป็น (เช่น เซิร์ฟเวอร์หรือบริการคลาวด์ ฐานข้อมูล ระบบสำรองข้อมูล และใบรับรอง SSL เพื่อความปลอดภัย) ให้พร้อมใช้งาน
- **การส่งมอบซอร์สโค้ด:** เมื่อพัฒนาระบบเสร็จสมบูรณ์ ทีมพัฒนาต้องส่งมอบ **ซอร์สโค้ดทั้งหมด** ของระบบให้แก่ผู้ว่าจ้าง รวมถึงโค้ดฝั่งเซิร์ฟเวอร์ โค้ดฝั่งไคลเอนต์ (LIFF PWA) และสคริปต์ที่เกี่ยวข้องกับโครงสร้างพื้นฐาน (Infrastructure as Code ถ้ามี)
- **เอกสารประกอบ:** ทีมพัฒนาต้องจัดทำเอกสาร **คู่มือการใช้งานระบบ** (สำหรับผู้ดูแลระบบและสำหรับผู้ใช้งานทั่วไป หากจำเป็น) และ **คู่มือการดูแลบำรุงรักษาระบบ** (Maintenance Manual) ที่อธิบายการติดตั้ง deployment ขั้นตอนการตั้งค่าที่สำคัญ วิธีการดูแลระบบ และการแก้ไขปัญหาเบื้องต้น เพื่อส่งมอบพร้อมกับซอร์สโค้ด
- **มาตรฐานคุณภาพ:** ระบบควรพัฒนาด้วยแนวทางที่ปลอดภัย มีเสถียรภาพ และปรับขยาย (scalable) ในอนาคตได้ง่าย ใช้เทคโนโลยีที่เหมาะสมและทันสมัย โดยให้คำปรึกษา/อัปเดตผู้ว่าจ้างเกี่ยวกับเทคโนโลยีที่จะใช้เป็นระยะ เพื่อให้มั่นใจว่าระบบจะรองรับการใช้งานในระยะยาว

5. การรองรับการขยายระบบในอนาคต

ระบบจะต้องถูกออกแบบโดยคำนึงถึง **การเติบโตและการขยายตัวในอนาคต** เพื่อรองรับความต้องการทางธุรกิจที่อาจเปลี่ยนแปลงหรือเพิ่มขึ้น ได้แก่:

- **เพิ่มสนามหรือสาขาใหม่:** ผู้ดูแลระบบ (Admin) ควรสามารถเพิ่มข้อมูล **สนามกีฬา/สถานที่จัดบริการใหม่** เข้าไปในระบบได้เองผ่านหน้าหลังบ้าน เช่น เพิ่มสาขาใหม่ที่มีสนามจำนวน X สนาม พร้อมตั้งค่าตารางเวลา และรายละเอียดอื่นๆ โดยไม่ต้องแก้ไขโค้ด
- **เพิ่มแพ็กเกจโปรโมชั่น:** ระบบควรรองรับการเพิ่ม **รแพ็กเกจโปรโมชั่นใหม่ๆ** ได้ง่าย ผู้ดูแลสามารถสร้าง/แก้ไข **แพ็กเกจเสริม** หรือโปรโมชั่นต่างๆ ผ่านหลังบ้าน และระบบจะแสดงผลให้ผู้ใช้เลือกซื้อได้ทันที
- **เพิ่มผู้ใช้งานและพนักงาน:** รองรับการเพิ่ม **บัญชีผู้ใช้หลังบ้าน** ในบทบาทต่างๆ เช่น เพิ่มพนักงานสาขาใหม่ เพิ่มผู้จัดการ หรือเพิ่มผู้ดูแลระบบส่วนกลาง ได้ไม่จำกัดตามสิทธิ์ที่กำหนด รวมถึงการถอน/ระงับสิทธิ์เมื่อพนักงานลาออกหรือเปลี่ยนหน้าที่
- **ปรับเปลี่ยนราคาและโปรโมชั่น:** ผู้ดูแลระบบสามารถ **ปรับปรุงราคา** ค่าบริการต่างๆ (ค่าเช่าสนาม) และ **ปรับเงื่อนไขโปรโมชั่น** ได้เองผ่านระบบหลังบ้าน (เช่น เปลี่ยนแปลงส่วนลด วัน-เวลาโปรโมชั่น) โดยระบบจะนำค่าที่แก้ไขใหม่ไปใช้ทันทีในการจอง/ขายครั้งต่อไป
- **รองรับโมดูลใหม่ในอนาคต:** โครงสร้างซอฟต์แวร์ควรมีความ **ยืดหยุ่น (flexible)** และ **เป็นโมดูลาร์ (modular)** ทำให้สามารถเพิ่มฟีเจอร์ใหม่ๆ (เช่น ระบบสมาชิกสะสมแต้ม ระบบวิเคราะห์ข้อมูลเชิงลึก ฯลฯ) เข้าไปภายหลังได้โดยไม่กระทบกับส่วนหลักของระบบที่พัฒนามาแล้ว หรือถ้ามีก็กระทบน้อยที่สุด

6. การเชื่อมต่อกับระบบภายนอก (Integration Requirements)

ในการพัฒนาระบบนี้ จะต้องมีการเชื่อมต่อกับระบบและบริการภายนอกต่างๆ เพื่อให้การทำงานครบวงจรและอัตโนมัติ ดังนี้:

- **เชื่อมต่อระบบบัญชี FlowAccount:** ระบบจะต้องเชื่อมต่อกับ **FlowAccount API** เพื่อบันทึกข้อมูลธุรกรรมทางการเงินต่างๆ โดยอัตโนมัติ เช่น เมื่อมีการเติมเงินเข้า Wallet (กระเป๋าเงิน) หรือมีการใช้เครดิต ระบบจะส่งรายการดังกล่าวไปยังระบบบัญชี FlowAccount เพื่อบันทึกเป็นรายการบัญชี (เช่น บันทึกเป็นหนี้สินเมื่อมีการเติมเงินเข้า wallet และตัดเป็นรายได้เมื่อผู้ใช้ใช้เครดิตชำระค่าบริการ) ซึ่งการเชื่อมต่อจะมาแทนที่การใช้ระบบ Peak ที่เคยพิจารณา
- **เชื่อมต่ออุปกรณ์ Igloohome (Deadbolt 2S):** ระบบจะต้องเชื่อมต่อกับ **Igloohome API** เพื่อจัดการอุปกรณ์ล็อกประตูแบบดิจิทัล โดยเมื่อมีการจองสนาม ระบบสามารถ **สร้างรหัส PIN** สำหรับกุญแจ Igloohome Deadbolt 2S ตามช่วงวัน-เวลาที่จอง และส่งให้ผู้ใช้งาน และเมื่อการจองสิ้นสุดลง ระบบสามารถ **ลบรหัส PIN** นั้นออกจากระบบล็อกโดยอัตโนมัติ การดำเนินการทั้งหมดนี้ต้องทำอย่างปลอดภัยและผู้จองจะเห็น PIN เฉพาะช่วงเวลาที่ได้รับสิทธิ์เข้าใช้สนามเท่านั้น
- **ระบบชำระเงิน OPN/Omise:** ระบบจะผนวกการชำระเงินออนไลน์ผ่าน **OPN Payments (Omise)** เพื่อรองรับการทำธุรกรรมต่างๆ ทั้งการเติมเงินเข้า Wallet และการชำระเงินรายรายการ (ในกรณีที่อนุญาต) โดยรองรับวิธีการชำระที่หลากหลาย เช่น พร้อมเพย์ (PromptPay QR), หักบัญชีธนาคาร (Direct Debit), บัตรเครดิต/เดบิต (Visa, Mastercard ทั้งในประเทศและต่างประเทศ) การเชื่อมต่อกับ Omise จะรวมถึงการรับ webhook ยืนยันสถานะการชำระเงินและการจัดการธุรกรรมล้มเหลวต่างๆ อย่างครบถ้วน
- **บริการ LINE API:** การทำงานหลายส่วนจะอาศัย **LINE API** ได้แก่ LINE Login (สำหรับรับรองตัวบุคคลผู้ใช้) และ LINE Messaging API (สำหรับส่งข้อความแจ้งเตือนผู้ใช้) ซึ่งระบบจะต้องเรียกใช้งาน API เหล่านี้ตามกรณีการใช้งานที่ระบุไว้ โดยปฏิบัติตามข้อกำหนดการใช้งานของ LINE อย่างเคร่งครัด

7. รายละเอียดโมดูลและฟังก์ชันระบบ (Functional Modules)

ด้านล่างนี้เป็นรายละเอียดของโมดูลหลักทั้ง 7 ส่วนที่ระบบต้องมี โดยแต่ละส่วนจะอธิบายขอบเขตการทำงานและคุณสมบัติสำคัญ:

7.1 ระบบจองสนาม (Court Booking System)

ระบบนี้เป็นหัวใจหลักที่ช่วยให้ผู้ใช้สามารถค้นหาและจองสนามกีฬาในสาขาที่ต้องการ พร้อมรับรหัสเข้าใช้สนาม (PIN) ผ่าน LINE โดยมีความสามารถดังนี้:

- **เลือกสถานที่ (Venue):** ผู้ใช้สามารถเลือกสาขาหรือสถานที่ที่ต้องการจะใช้บริการได้ หากผู้ใช้เปิดใช้งาน Location/GPS ระบบอาจช่วยค้นหาสาขาที่อยู่ใกล้ผู้ใช้ที่สุดโดยอัตโนมัติ แต่ผู้ใช้ยังสามารถปรับเปลี่ยนเลือกสถานที่เองได้
- **เลือกวันและเวลา:** ผู้ใช้เลือกวันที่ต้องการจอง และช่วงเวลาเริ่มต้น รวมถึงระยะเวลาที่ต้องการใช้สนาม (ระบบควรกำหนดตัวเลือกมาตรฐาน เช่น ครั้งละ 60 นาที หรือ 90/120 นาที เป็นต้น โดยแสดงในรูปแบบ Time Grid ช่วงละ 30 นาที เพื่อความละเอียดของช่วงเวลาที่จองได้)

- **ตรวจสอบสนามที่ว่าง:** เมื่อระบุสถานที่และเวลาที่ต้องการ ระบบจะแสดงรายการ **สนาม (คอร์ท)** ที่ว่างให้จองได้ในช่วงเวลาดังกล่าว โดยแสดงในรูปแบบที่เข้าใจง่าย เช่น **ปฏิทินหรือผังเวลาที่เป็นภาพ (visual calendar)** เพื่อให้ผู้ใช้เห็นภาพรวมว่าสนามใดบ้างว่าง/ไม่ว่าง
- ระบบต้องคำนึงถึง **ปัจจัยการปิดสนาม** เช่น ช่วงเวลาที่สนามถูกจองไปแล้ว, ช่วงเวลาปิดปรับปรุง หรือ Buffer Time ก่อน/หลังการใช้สนาม (ถ้ามีนโยบายให้พักสนามระหว่างรอบการจอง)
- หากสถานที่นั้นมีหลายสนาม ระบบควรแสดง **แผนผังสนาม** (เช่น Court 1, Court 2, ... หรือ indoor/outdoor) เพื่อให้ผู้ใช้เลือกได้ตรงความต้องการ และในกรณีที่ต้องการจองหลายสนามพร้อมกัน (เช่น กรณีจัดแข่งหรือเล่นเป็นกลุ่มใหญ่) ควรสามารถเลือกได้หลายสนามในขั้นตอนเดียวกัน
- **การเลือกบริการเสริม (Add-ons):** ก่อนทำการยืนยันการจอง ผู้ใช้สามารถเลือก **บริการหรือสินค้าเสริม** เพิ่มเติมไปพร้อมกับการจองสนามได้ เช่น สั่งจองโค้ชหรือเทรนเนอร์ส่วนตัวสำหรับเวลาที่จอง, สั่งจองผ้าเช็ดตัวหรือเครื่องดื่มที่จะเตรียมไว้ให้, เช่าอุปกรณ์การเล่น (เช่น ไม้ปิงปอง ลูกบอล) หรือเลือกแพ็กเกจโปรโมชั่นที่มี (ดูรายละเอียดระบบแพ็กเกจในหัวข้อ 7.2) การเลือก Add-ons จะสะดวกต่อผู้ใช้และช่วยเพิ่มรายได้ต่อการจอง
- **การคำนวณค่าใช้จ่าย:** ระบบจะคำนวณ **ค่าบริการรวม** ของการจองนั้นโดยรวมค่าจองสนามตามระยะเวลาและประเภทสนาม, ค่าบริการเสริม/สินค้าเพิ่มเติมที่ผู้ใช้เลือก, รวมถึงภาษีหรือค่าธรรมเนียมอื่นๆ (ถ้ามี) แล้วแสดงให้ผู้ใช้ตรวจสอบก่อนยืนยันการจอง
- **การเติมเงิน/ชำระเงิน:** เมื่อตกลงจะจอง ผู้ใช้จะเข้าสู่ขั้นตอน **ชำระเงิน (Checkout)** โดยระบบจะเน้นการหักเงินจาก **กระเป๋าเงิน (Wallet)** ของผู้ใช้:
- หากยอดเงินใน Wallet ของผู้ใช้ **เพียงพอ** ต่อค่าบริการที่ต้องชำระ ระบบจะสามารถหักเงินนั้นได้ทันที
- หาก **ยอดไม่พอ** ระบบจะแจ้งเตือนและให้ผู้ใช้ทำการ **เติมเงินเข้า Wallet** ก่อน (ผ่านการชำระด้วยช่องทางที่รองรับในระบบ Wallet & Payment ดูหัวข้อ 7.4) เมื่อเติมเสร็จแล้วจึงทำรายการต่อ
- *หมายเหตุ:* การชำระเงินโดยตรงด้วยบัตรเครดิตหรือวิธีอื่นนอกรูปแบบ Wallet อาจไม่จำเป็นต้องรองรับ เนื่องจากผู้ใช้ทุกคนมีบัญชี LINE และ Wallet อยู่แล้ว (นโยบายคือใช้ LINE Login เท่านั้น)
- **ยืนยันการจอง & การสร้าง PIN:** เมื่อชำระเงินเรียบร้อย ระบบจะทำการ **ยืนยันการจอง** พร้อมสร้าง **รหัส PIN สำหรับเข้าพื้นที่สนาม** ที่จองไว้:
- ระบบจะเรียกใช้งาน Igloohome API เพื่อสร้าง PIN 4-6 หลัก (ขึ้นกับอุปกรณ์) ให้แก่ **ประตู/กุญแจของสนามนั้น** โดยกำหนดช่วงเวลาให้ PIN นี้ใช้งานได้ตรงกับเวลาที่จอง (เช่น เริ่มใช้ได้ 10 นาทีก่อนเวลาจอง และหมดอายุหลังเวลาจองสิ้นสุดไม่เกิน 10 นาที) เพื่อให้ผู้ใช้นำไปใช้ปลดล็อกเข้าพื้นที่
- **การส่ง PIN:** ระบบจะส่ง PIN นี้ให้ผู้จองผ่านทาง **ข้อความ LINE OA** โดยทันทีที่จองเสร็จในรูปแบบที่ **ปลอดภัย** (เช่น ส่งเป็นข้อความบอกว่าการจองสำเร็จ และมีปุ่ม/ลิงก์ “ดูรายละเอียดการจอง” ซึ่งเมื่อกดแล้วเปิด LIFF เพื่อแสดง PIN ภายในแอป แทนการส่งตัวเลข PIN ตรงๆ ในแชท)
- ผู้จองยังสามารถเลือก **แชร์ลิงก์รายละเอียดการจองพร้อม PIN** นี้ไปให้เพื่อนที่ร่วมเล่นได้ผ่าน LINE เช่นกัน (ระบบจะสร้างลิงก์เฉพาะบุคคล โดยเมื่อเพื่อนกดลิงก์จะถูกให้ล็อกอินผ่าน LINE เพื่อยืนยันตัวตนก่อน และสามารถเห็นรายละเอียดการจองและ PIN ได้ แต่ **PIN จะไม่ถูกเผยแพร่ในแชทสาธารณะ** เพื่อป้องกันการเข้าถึงโดยผู้ไม่มีสิทธิ์)

- **การใช้งาน PIN อย่างปลอดภัย:** PIN ที่ออกให้แต่ละครั้งจะผูกกับการจองนั้นๆ และมี **ข้อจำกัด** เพื่อความปลอดภัย เช่น ใช้ได้ครั้งเดียวหรือใช้ได้เฉพาะช่วงเวลาที่กำหนดเท่านั้น, ถูก **เข้ารหัสหรือแฮช (Hash)** เมื่อเก็บในฐานข้อมูล (ไม่มีการเก็บรหัส PIN แบบ plaintext), และมีการจำกัดจำนวนครั้งที่สามารถใช้ปลดล็อก (เช่น ใช้ได้ 1 ครั้ง หรือจำกัดจำนวนครั้งภายในเวลาจอง)
- หลังผู้ใช้เข้าใช้สนามและสิ้นสุดเวลา ระบบสามารถ **ยกเลิก PIN** นั้นโดยอัตโนมัติ (ผ่าน Igloohome API) เพื่อความปลอดภัย โดยผู้ใช้จะไม่สามารถใช้ PIN ซ้ำในเวลาที่ไม่ใช่เวลาจอง

7.2 ระบบแพ็กเกจและบริการเสริม (Package & Add-on System)

โมดูลนี้ช่วยให้ธุรกิจสามารถนำเสนอข้อเสนอพิเศษหรือบริการเพิ่มเติมเพื่อเพิ่มมูลค่าให้กับลูกค้า และช่วยให้ผู้ดูแลสามารถจัดการรายการแพ็กเกจ/บริการเสริมต่างๆ ได้สะดวก:

- **การจัดการแพ็กเกจโปรโมชั่น:** ผู้ดูแลระบบสามารถสร้างและบริหาร **แพ็กเกจ** ที่รวมหลายรายการเข้าด้วยกัน เช่น แพ็กเกจจัดอีเวนต์ที่รวมการจองสนาม 2 ชั่วโมง + โค้ชส่วนตัว + เครื่องดื่มสำหรับผู้เล่น หรือแพ็กเกจเล่นเป็นกลุ่มที่คิดค่าบริการตามจำนวนผู้เข้าร่วม เป็นต้น โดยแต่ละแพ็กเกจสามารถกำหนด:
- **ประเภทของทรัพยากรที่รวมในแพ็กเกจ:** เช่น จำนวนสนามที่จะจอง, จำนวนโค้ช/เทรนเนอร์, รายการอาหาร/เครื่องดื่ม หรือสินค้า merchandise ที่จะได้รับ เป็นต้น
- **ราคาและเงื่อนไขการคิดเงิน:** สามารถตั้งเป็นราคาคงที่ของแพ็กเกจ หรือกำหนดสูตรคำนวณตามปัจจัย (เช่น คิดตามจำนวนผู้เข้าร่วม หรือจำนวนสนามที่ใช้) เช่น สูตรคำนวณราคา = (จำนวนสนาม * อัตราค่าสนามต่อสนาม) + (จำนวนผู้เล่น * ราคาต่อหัว) + ราคาเหมาจ่ายของบริการเสริมอื่นๆ
- **ระยะเวลาโปรโมชั่น:** กำหนดช่วงวันที่ที่แพ็กเกจนี้มีผล หรือช่วงเวลาที่งดให้บริการ (blackout dates) เช่น แพ็กเกจบางอย่างอาจให้บริการเฉพาะวันธรรมดา หรือยกเว้นวันหยุดนักขัตฤกษ์
- **รายละเอียดอื่นๆ:** เช่น รูปภาพประกอบแพ็กเกจ คำอธิบายเงื่อนไข สิทธิประโยชน์ที่จะได้รับ เพื่อแสดงในหน้าให้ผู้ใช้เลือก
- **บริการเสริม (Add-ons):** นอกจากแพ็กเกจแบบรวมแล้ว ระบบควรรองรับการเพิ่ม **บริการ/สินค้าเสริมรายรายการ** ที่ผู้ใช้สามารถเลือกเพิ่มเข้ามาในการจองหรือคำสั่งซื้อของตน เช่น การเช่าโค้ชส่วนตัวต่อชั่วโมง, การเช่าอุปกรณ์กีฬาเพิ่มเติม (นอกเหนือจากที่รวมในค่าจองสนาม), การสั่งชุดอาหาร/เครื่องดื่มต้อนรับ เป็นต้น โดยบริการเสริมเหล่านี้ ผู้ดูแลระบบสามารถเพิ่ม/แก้ไขได้ผ่านหลังบ้านและกำหนดราคาได้ตามต้องการ
- **การแสดงผลแก่ผู้ใช้:** ในขั้นตอนการจองหรือสั่งซื้อ ระบบจะแสดงตัวเลือก **แพ็กเกจโปรโมชั่น** ที่มีอยู่ และรายการ **บริการเสริม/สินค้าเสริม** ที่เกี่ยวข้องให้ผู้ใช้เลือกเพิ่มเติม (เช่น หลังเลือกสนามและเวลา อาจมีตัวเลือก “แพ็กเกจปาร์ตี้ 8 คน” หรือ “เพิ่มโค้ชส่วนตัว 1
-
- **ผลลัพธ์หลังการจองด้วยแพ็กเกจ:** หากผู้ใช้เลือกแพ็กเกจที่มีสิทธิพิเศษหรือของแถม ระบบต้องจัดการ **สิทธิประโยชน์** เหล่านั้นให้ครบถ้วน เช่น หากแพ็กเกจรวมเครื่องดื่ม ระบบ POS/Kitchen ควรรับรู้รายการนี้, หากรวมโค้ช ระบบควรมีข้อมูลให้ผู้จัดการสาขาจัดตารางโค้ชให้ตรงเวลาที่จอง เป็นต้น รวมถึงแสดงรายละเอียดสิทธิ์ในใบยืนยันการจองของผู้ใช้อย่างชัดเจน

7.3 ระบบโปรโมชั่นและสิทธิพิเศษ (Promotions & Specials System)

โมดูลนี้ช่วยให้ธุรกิจสามารถตั้งค่า **โปรโมชั่นการตลาด** และ **ข้อเสนอพิเศษ** ต่างๆ เพื่อจูงใจลูกค้าและเพิ่มยอดขาย โดยระบบจะต้องยืดหยุ่นพอที่จะรองรับเงื่อนไขโปรโมชั่นที่หลากหลาย:

- **ประเภทของโปรโมชั่นที่รองรับ:**
- **ส่วนลดตามช่วงวัน/เวลา (Happy Hour):** เช่น ตั้งโปรโมชันให้การจองสนามในวันธรรมดา (จันทร์-ศุกร์) ก่อน 16:00 น. ได้รับส่วนลด 20% หรือช่วงเวลาที่ลูกค้าใช้น้อยให้ส่วนลดพิเศษ เป็นต้น
- **ส่วนลดตามเงื่อนไขลูกค้า:** เช่น ส่วนลดสำหรับลูกค้าใหม่ครั้งแรก, ส่วนลดวันเกิด, หรือส่วนลดสำหรับสมาชิก (ในอนาคตเมื่อมีระบบสมาชิก)
- **โปรโมชั่นแบบ Bundle/แพ็กเกจ:** ซื้อหรือจองเป็นชุดแล้วได้ราคาพิเศษ เช่น จองสนาม 10 ชั่วโมงล่วงหน้าในราคาเหมารวม, ซื้อเครื่องดื่ม + เช่าสนามคู่กันได้ราคาถูกลง, หรือ “เล่น 10 ครั้ง ฟรี 1 ครั้ง” เป็นต้น
- **คูปองหรือโค้ดส่วนลด:** สร้างรหัสโปรโมชั่นที่ลูกค้ากรอกตอนจองเพื่อรับสิทธิ์ (ระบบควรรองรับ input โค้ดส่วนลดและตรวจสอบความถูกต้อง/เงื่อนไข)
- **การตั้งค่าเงื่อนไขแบบไดนามิก:** ผู้ดูแลระบบสามารถกำหนด **เงื่อนไข** ของโปรโมชั่นได้หลากหลายและซับซ้อน โดยผ่านหน้า UI ที่เข้าใจง่าย เช่น
- เลือกประเภทโปรโมชั่น (ส่วนลดเป็นเปอร์เซ็นต์, ส่วนลดเป็นจำนวนเงิน, ของแถม, ฯลฯ)
- ระบุ **เงื่อนไขการใช้:** วันในสัปดาห์, ช่วงเวลาในวัน, ประเภทสนามหรือสินค้าที่ร่วมรายการ, ยอดขั้นต่ำที่ต้องซื้อ, จำนวนครั้งที่ใช้ได้ต่อผู้ใช้, ฯลฯ
- ระบุ **ช่วงเวลาที่โปรโมชั่นมีผล:** วันที่เริ่มต้น-สิ้นสุด, หรือกำหนดเป็นช่วงเวลาในแต่ละวัน
- ระบบจะใช้เงื่อนไขเหล่านี้ในการคำนวณและตรวจสอบ **แบบอัตโนมัติ** เมื่อผู้ใช้ทำรายการ เช่น ถ้าตรงเงื่อนไขก็จะลดราคาให้เลย หรือแจ้งเตือนผู้ใช้ถึงโปรโมชั่นที่ได้รับ
- **การแสดงผลและใช้งาน:** เมื่อมีโปรโมชั่นที่เกี่ยวข้อง ระบบควรแจ้งหรือแสดงให้ผู้ใช้ทราบในขั้นตอนที่เกี่ยวข้อง เช่น
- แสดงป้าย “ลดราคา” หรือราคาใหม่ที่ลดแล้วในเมนูสินค้า/บริการที่ร่วมโปร
- ในหน้าสรุปการชำระเงิน (Checkout) แสดงรายการโปรโมชั่นที่ถูกปรับใช้ และจำนวนเงินที่ลดให้เห็นชัดเจน
- มีหน้ารวม **รายการโปรโมชั่นที่มีอยู่** ในขณะนั้น เพื่อให้ลูกค้ารับทราบ (เช่น เมนู “โปรโมชั่น” ใน LIFF)
- **ตัวอย่าง:**
- ตั้งโปรโมชั่น “**Weekend Special**” ลดค่าเช่าสนาม 15% สำหรับการจองที่เล่นวันเสาร์-อาทิตย์
- ตั้งโปรโมชั่น “**Happy Hour**” ลดราคาอาหาร/เครื่องดื่ม 10% สำหรับการสั่งในช่วง 14:00-16:00 น. จันทร์-ศุกร์
- ตั้ง “**โปรชวนเพื่อน**” ลูกค้าปัจจุบันที่แนะนำเพื่อนใหม่มาสมัครและจองสนาม ทั้งคู่จะได้รับเครดิตเพิ่มคนละ 100 บาท (อันนี้อาจต้องมีระบบ referral ซัพพอร์ตในอนาคต)
- **ระบบควบคุมและจำกัด:** ควรมีการจำกัดจำนวนหรือการควบคุมการใช้โปรโมชั่นบางประเภท เพื่อป้องกันการใช้งานเกินสิทธิ์ เช่น จำกัดจำนวนคูปองทั้งหมดที่มี, จำกัด 1 คนใช้ได้กี่ครั้ง, หรือให้โปรโมชั่นสิ้นสุดเมื่อครบจำนวนสิทธิ์ที่กำหนด

7.4 ระบบกระเป๋าเงิน (Wallet) และการชำระเงิน (Payment)

ระบบกระเป๋าเงินดิจิทัลเป็นส่วนสำคัญที่ใช้ในการเก็บมูลค่า (เครดิต) ของลูกค้าและใช้สำหรับชำระค่าบริการต่างๆ ในแพลตฟอร์ม โดยผนวกกับระบบการรับชำระเงินจากภายนอกอย่าง OPN/Omise:

- **กระเป๋าเงินดิจิทัล (Wallet – "District Credits"):** ผู้ใช้แต่ละคนจะมี **บัญชีกระเป๋าเงิน** ในระบบ ผูกกับบัญชี LINE ของตน ซึ่งแสดงยอดเงินคงเหลือ (credits) ที่สามารถใช้จ่ายได้ภายในแพลตฟอร์ม
- ผู้ใช้สามารถเติมเงินเข้า Wallet นี้ด้วยการชำระผ่านช่องทางที่กำหนด (เช่น บัตรเครดิต/เดบิต, PromptPay ฯลฯ) ยอดเงินจะถูกแปลงเป็นเครดิตในอัตรา 1 บาท = 1 credit และสะสมในบัญชี
- **ไม่มีการคืนเงินสด**: เครดิตที่เติมเข้าสู่ Wallet จะไม่สามารถแลกคืนเป็นเงินสดได้ (non-refundable) ผู้ใช้จำเป็นต้องใช้เครดิตเหล่านี้ไปกับบริการ/สินค้าภายในระบบเท่านั้น
- **อายุของเครดิต:** เครดิตมีอายุการใช้งาน **12 เดือน** นับจากวันที่เติม (หรือตามนโยบายที่กำหนด) หากเกินกำหนดจะหมดอายุและถูกตัดออกจากบัญชี โดยระบบจะแจ้งเตือนผู้ใช้ล่วงหน้าก่อนเครดิตหมดอายุ (เช่น แจ้งเตือนผ่าน LINE ก่อน 3 เดือน, 2 เดือน, 1 เดือนตามลำดับ)
- **การเติมเงิน (Top-up):**
- ผู้ใช้สามารถเลือกแพ็กเกจเติมเงินมาตรฐาน เช่น เติม 1,000 บาท, 5,000 บาท หรืออาจมีโปรโมชั่นการเติม (เช่น เติม 10,000 บาท ได้รับ 12,000 credits เพื่อจูงใจ)
- เมื่อผู้ใช้เลือกจำนวนเงินที่จะเติม ระบบจะแสดงยอดเงินที่ต้องชำระและเรียกไปยัง **OPN/Omise** เพื่อดำเนินการตัดบัตรหรือช่องทางที่เลือก เมื่อได้รับยืนยันสำเร็จจาก Omise ระบบจะเพิ่มเครดิตเข้ากระเป๋าเงินของผู้ใช้ตามจำนวนที่เติม
- ควรมีกลไกป้องกันการเติมเงินซ้ำซ้อนหรือข้อผิดพลาด (เช่น ถ้าผู้ใช้ปิดหน้าต่างกลางคัน ต้องตรวจสอบกับ Omise ก่อนเพิ่มเครดิตว่าการชำระเสร็จสมบูรณ์จริง)
- **การใช้เครดิต/ชำระเงิน:** ในการทำธุรกรรมภายในระบบ (เช่น จองสนาม สั่งอาหาร ซื้อสินค้า) ระบบจะใช้เครดิตใน Wallet ของผู้ใช้เป็นหลักในการชำระ:
- หากเครดิตเพียงพอกับยอดที่ต้องชำระ ระบบจะหักเครดิตและยืนยันรายการได้ทันที
- หากเครดิตไม่พอ ระบบจะแจ้งผู้ใช้ให้เติมเงินเพิ่ม (top-up) ให้เกินหรือเท่ากับยอดที่ต้องชำระก่อน จึงจะดำเนินการต่อได้
- **กฎขั้นต่ำการชำระเงิน:** หากมีการชำระผ่านช่องทางบัตรหรือบัญชีธนาคารโดยตรง (กรณีเติมเงินหรือชำระเงินอื่นๆ) กำหนดขั้นต่ำแต่ละช่องทาง เช่น
    - PromptPay: ไม่กำหนดยอดขั้นต่ำ
    - Direct Debit (หักบัญชีธนาคาร): ขั้นต่ำ 400 บาทต่อรายการ
    - บัตรเครดิต/เดบิต (Domestic): ขั้นต่ำ 600 บาทต่อรายการ
    - บัตรเครดิต (International): ขั้นต่ำ 600 บาทต่อรายการ และมีค่าธรรมเนียมเพิ่มเติมครั้งละ 15 บาทต่อการใช้บัตรต่างประเทศ (ระบบต้องตรวจสอบจากเลข BIN ของบัตรและบวกค่าธรรมเนียมนี้เข้าไปในการทำรายการโดยอัตโนมัติ)
- ตัวอย่าง: ถ้าผู้ใช้จะเติมเงิน 300 บาท ระบบควรแจ้งว่าสามารถใช้ PromptPay ได้ แต่ไม่สามารถใช้ Direct Debit หรือบัตรเครดิต (เนื่องจากต่ำกว่ายอดขั้นต่ำ)
- **ฟีเจอร์พิเศษของ Wallet:**
- **Gift Credits:** ผู้ใช้สามารถ **มอบเครดิตให้ผู้อื่น** ได้ เช่น มีฟังก์ชัน "เติมเครดิตให้เพื่อน" ซึ่งจะหักเครดิตจากผู้ให้และเพิ่มให้บัญชี Wallet ของผู้รับ (อ้างอิงจาก LINE ID) โดย

กระบวนการนี้ต้องมีการยืนยันตัวตนและความปลอดภัย เพื่อป้องกันการโอนผิดบัญชี (หมายเหตุ: การโอนเครดิตระหว่าง Wallet ผู้ใช้โดยตรงอาจไม่อนุญาต แต่ทำได้ในรูปแบบของการ gifting เป็นครั้งๆ)

- **บัตรของขวัญ (Gift Card):** ในระบบ merchandise อาจมี SKU พิเศษที่แทนบัตรของขวัญ ซึ่งเมื่อซื้อแล้วสามารถส่งเป็นของขวัญให้ผู้อื่นและเติมเข้าวอลเล็ตของเขาได้
- **การแยกยอดเติมเงิน:** สำหรับการใช้งาน UX ที่ง่าย ควรแยก flow “เติมเงินล่วงหน้า” ออกจาก “การจ่ายเงินตอน checkout” อย่างชัดเจน เพื่อป้องกันความสับสน เช่น ผู้ใช้ควรเติมเงินเข้า Wallet ก่อน แล้วค่อยกลับมาจ่ายค่า booking แทนที่จะพยายามรวมขั้นตอน (แต่ระบบสามารถมี prompt ลัดให้ในขั้นตอน checkout เพื่อให้เติมเงินทันทีได้)
- **ระบบใบเสร็จและประวัติการทำรายการ:** ผู้ใช้สามารถดู **ประวัติการเติมเงิน** และ **การใช้จ่าย** ของ Wallet ตนเองได้ทั้งหมด (Transaction History) ในหน้าโปรไฟล์ เช่น รายการเติมเงิน, รายการหักจ่ายแต่ละครั้ง ระบุวันที่ เวลา และรายละเอียด โดยสามารถกดดูใบเสร็จดิจิทัลของแต่ละรายการได้
- **ความปลอดภัย:** ระบบจะไม่จัดเก็บข้อมูลบัตรเครดิตของผู้ใช้โดยตรง (ทุกการกรอกข้อมูลบัตรจะทำผ่านแบบฟอร์มของ OPN/Omise หรือวิธีที่เป็น PCI-compliant) และข้อมูลสำคัญจะถูกส่งตรงไปยังผู้ให้บริการการชำระเงิน

7.5 ระบบหลังบ้าน (Admin Dashboard)

ระบบหลังบ้านเป็นเครื่องมือให้ผู้ดูแลระบบและพนักงานที่มีสิทธิ์เข้าถึงเพื่อจัดการการดำเนินงานของแพลตฟอร์ม ซึ่งต้องออกแบบให้ใช้งานง่ายและครอบคลุมฟังก์ชันที่จำเป็นดังนี้:

- **การจัดการผู้ใช้และสิทธิ์ (User Roles & Permissions):** รองรับการสร้างและบริหาร **บัญชีผู้ใช้หลังบ้าน** ในหลายระดับสิทธิ์ ได้แก่:
- **Admin (ส่วนกลาง):** บทบาทผู้ดูแลระบบระดับสูงสุด สามารถเข้าถึงทุกเมนูและข้อมูลทั่วทั้งแพลตฟอร์ม ทำการตั้งค่าระบบระดับนโยบาย (เช่น กฎการชำระเงิน ขั้นต่ำ/ค่าธรรมเนียม, การกำหนดแพ็กเกจโปรโมชั่นกลาง, การเชื่อมต่อบัญชี/ระบบภายนอกอย่าง FlowAccount) จัดการข้อมูลทุกสาขา ดูรายงานรวม และมีสิทธิ์ในการอนุมัติ/แก้ไขธุรกรรมต่างๆ รวมถึงการจัดการบัญชีผู้ใช้หลังบ้านคนอื่นๆ (เพิ่ม/ลดสิทธิ์)
- **Venue Manager (ผู้จัดการสาขา):** บทบาทสำหรับผู้จัดการหรือผู้รับผิดชอบประจำสถานที่แต่ละแห่ง สามารถจัดการข้อมูลเฉพาะสาขาของตัวเองได้ เช่น ตั้งค่ารายละเอียดสถานที่และสนาม (เวลาเปิดปิด, ปิดปรับปรุง), เมนูอาหาร/เครื่องดื่มของสาขา, ข้อมูลสินค้าคงคลังของสาขา, จัดการบัญชีพนักงานของสาขาตัวเอง และดูรายงานการขาย/การจองที่เกิดในสาขานั้นๆ
- **Staff (พนักงานทั่วไป, POS/KDS):** บทบาทสำหรับพนักงานปฏิบัติงาน เช่น พนักงานหน้าร้านหรือพนักงานครัว สิทธิ์จะจำกัดเฉพาะฟังก์ชันที่เกี่ยวข้อง เช่น เข้าหน้าจอ POS เพื่อลงคำสั่งซื้อและรับชำระเงิน, เข้าหน้าจอ Kitchen Display เพื่อดูรายการอาหารที่ต้องเตรียม, อนุมัติคืนเงิน/ยกเลิกรายการเฉพาะที่เกี่ยวกับลูกค้าที่ตนรับผิดชอบ (เช่น ยกเลิกออเดอร์ถ้าลูกค้ายกเลิก)
- **Customer (ลูกค้า):** ไม่ใช่ผู้ใช้หลังบ้าน แต่เพื่อความสมบูรณ์ คือลูกค้าทั่วไปที่มีสิทธิ์ใช้เฉพาะผ่าน LIFF เท่านั้น ทำการจอง สั่งของ ใช้วอลเล็ต ดูใบเสร็จของตัวเอง
- **Investor (นักลงทุน/เจ้าของ):** (ถ้ามี) เป็นบทบาทสำหรับผู้บริหารหรือนักลงทุนที่ต้องการดูข้อมูลสรุปทางธุรกิจ โดยสิทธิ์นี้จะสามารถดู **รายงานสรุป** เช่น ผลประกอบการ (P&L)

ของแต่ละสาขาหรือรวมทุกสาขา และประวัติการเบิกจ่ายกำไร แต่จะไม่เห็นข้อมูลปลีกย่อยระดับรายการการจอง/คำสั่งซื้อ หรือจัดการอะไรไม่ได้ (read-only)

- **การจัดการสถานที่และสนาม:** ในหลังบ้านส่วนกลางหรือของแต่ละสาขาจะมีเมนูให้ **เพิ่ม/แก้ไขข้อมูลสถานที่และสนามกีฬา**:
- เพิ่มสาขาใหม่ แก้ไขชื่อ ที่อยู่ พิกัด สาขาที่มีอยู่
- เพิ่ม/แก้ไขสนาม (เช่น Court 1, Court 2...) พร้อมระบุคุณสมบัติ (ในร่ม/กลางแจ้ง, ประเภทพื้น, จำนวนคนที่รองรับ, ฯลฯ)
- กำหนด **เวลาทำการของสนาม** (เช่น เปิดทุกวัน 8:00-22:00 ยกเว้นวันหยุดนักขัตฤกษ์, หรือช่วงพักสนามระหว่างวัน)
- ตั้งค่า **ช่วงปิดสนาม (Blackout)** เช่น ปิดปรับปรุง, สนามไม่ว่างช่วงจัดอีเวนต์
- จัดการ **Layout/ชื่อเรียก** ของสนามเพื่อใช้แสดงผลในระบบจอง
- **การจัดการเมนูสินค้า/บริการ:** เมนูสำหรับ **สินค้าและบริการ** ที่มีให้ลูกค้า:
- เพิ่ม/แก้ไข **เมนูอาหาร/เครื่องดื่ม** ที่ขาย (พร้อมราคาขาย, ต้นทุนวัตถุดิบ, สูตร BOM ถ้ามี)
- เพิ่ม/แก้ไข **สินค้าขาย (Merchandise)** แต่ละรายการ (ชื่อ, ประเภท, ราคา, จำนวนคงเหลือ, รูปภาพ)
- เพิ่ม/แก้ไข **บริการเสริม/การเช่า** เช่น ประเภทโค้ชที่ให้บริการ, รายการอุปกรณ์ที่ปล่อยเช่าพร้อมค่าบริการหรือค่ามัดจำ
- จัดการ **โปรโมชั่น** (ตามหัวข้อ 7.3) เช่น สร้างโปรโมชั่นใหม่ เปิด/ปิดโปรโมชั่น แก้ไขเงื่อนไข
- **ระบบคลังสินค้า (Inventory Backend):** สำหรับผู้จัดการหรือแอดมินเพื่อ:
- บันทึกการ **เพิ่มสต็อก** สินค้า/วัตถุดิบเมื่อมีการนำเข้า (เช่น เมื่อของมาส่ง)
- บันทึกการ **ปรับปรุงสต็อก** กรณีมีของเสียหาย สูญหาย หรือตรวจนับแล้วไม่ตรง
- ดู **สถานะสต็อก** ของทุกรายการแบบเรียลไทม์ และรายการใดที่เหลือน้อยกว่าจุดสั่งซื้อ
- ออกรายงาน **บัญชีคงเหลือ** สินค้า
- **การจัดการคำสั่งและลูกค้า:** ส่วนนี้ช่วยให้แอดมิน/ผู้จัดการติดตามสถานการณ์:
- ดูรายการ **การจองสนาม** ทั้งหมดที่เข้ามา (filter ตามสาขา วันที่ สถานะ) สามารถดำเนินการบางอย่างได้ เช่น อนุมัติ/ยกเลิกการจอง (กรณีพิเศษ)
- ดูรายการ **คำสั่งซื้อ (Orders)** ของสินค้า/อาหาร ที่เกิดขึ้นผ่าน POS หรือผ่าน LIFF (ดูสถานะ เช่น กำลังเตรียม/สำเร็จ/ยกเลิก)
- เข้าดู **โปรไฟล์ลูกค้า** รายบุคคลได้ เช่น ประวัติการจอง จำนวนเครดิตที่มี หรือประวัติการเติมเงิน (ใช้สำหรับบริการลูกค้าเมื่อเกิดปัญหา)
- **ระบบรายงาน (Reporting):** หลังบ้านจะมีหน้ารายงานและส่งออกข้อมูลเพื่อใช้วิเคราะห์และทำบัญชี:
- **รายงานยอดขายรายวัน/รายเดือน** แยกตามประเภท (รายได้จากการจองสนาม, รายได้จาก F&B, รายได้จาก Merchandise, รายได้อื่นๆ) และสามารถแยกตามสาขา
- **รายงานการใช้สนาม** เช่น อัตราการถูกจองของแต่ละสนาม รายช่วงเวลา ฯลฯ
- **รายงานการเงิน/บัญชี:** เช่น รายงาน P&L (กำไรขาดทุน) ของแต่ละสาขารายเดือน (รายได้ - ต้นทุน), รวมถึงรายงานสรุปรวมสำหรับ HQ
- **การส่งออก Excel/CSV:** รายงานทั้งหมดควรสามารถกด **Export** เป็นไฟล์ Excel หรือ CSV ได้ เพื่อความสะดวกในการนำไปประมวลผลต่อภายนอก

- **ระบบบัญชี (FlowAccount):** ในส่วนหลังบ้านสำหรับบัญชี อาจมีปุ่มให้ **Export/Sync ข้อมูลไปยัง FlowAccount** (เช่น ข้อมูลการเติมเงิน wallet เพื่อบันทึกเป็น liability และข้อมูลการใช้ wallet เพื่อบันทึกเป็น revenue) ซึ่งอาจตั้งให้ซิงค์อัตโนมัติรายวันช่วงกลางคืน หรือกดส่งเองก็ได้ ทั้งนี้ต้องสร้างไฟล์หรือข้อมูลตามรูปแบบที่ FlowAccount ต้องการ
- **Audit Logs:** ระบบหลังบ้านควรเก็บ **บันทึกการทำงาน** (audit log) ของผู้ใช้หลังบ้านด้วย เช่น ใครเข้าระบบเมื่อไร แก้ไขข้อมูลใดไปบ้าง ลบรายการอะไรเมื่อไร เพื่อใช้ในการตรวจสอบกรณีเกิดข้อผิดพลาดหรือทุจริต

7.6 ระบบการสื่อสารแจ้งเตือน (LINE Push Notification System)

ระบบการแจ้งเตือนจะใช้ LINE OA ในการสื่อสารข้อมูลสำคัญไปยังผู้ใช้อย่างอัตโนมัติ สร้างประสบการณ์ที่ต่อเนื่องและลดการตกหล่นของข้อมูล:

- **ประเภทการแจ้งเตือนที่ระบบต้องส่ง:**
- **การยืนยันการจอง:** เมื่อมีการจองสนามสำเร็จ ระบบส่งข้อความถึงผู้ใช้ (ผ่านบัญชี LINE OA) เพื่อยืนยันรายละเอียดการจอง (วันที่/เวลา สถานที่ที่จอง หมายเลขสนาม ฯลฯ) พร้อมกับลิงก์สำหรับดู PIN หรือข้อมูลเข้าใช้สนาม
- **แจ้งรหัสเข้าพื้นที่ (PIN):** ส่ง PIN หรือวิธีการเข้าถึงสนาม (อาจเป็น QR ในอนาคต) ให้ผู้ใช้ผ่านช่องทางที่ปลอดภัย (โดยหลักการคือให้ผู้ใช้กดลิงก์เพื่อดูใน LIFF แทนการแสดง PIN ตรงๆ ในข้อความ)
- **แจ้งเตือนการชำระเงิน:** เมื่อผู้ใช้ทำการเติมเงินเข้า Wallet หรือมีการชำระเงิน/ตัดเครดิต ระบบส่งข้อความยืนยันการได้รับชำระเงินสำเร็จ พร้อมรายละเอียด (ยอดเงิน วันที่ เวลา หรือ transaction id)
- **สถานะคำสั่งซื้อ F&B:** กรณีผู้ใช้สั่งอาหาร/เครื่องดื่มผ่านระบบ (หรือสั่งผ่านพนักงานแล้วผูกกับบัญชีผู้ใช้) ระบบสามารถส่งสถานะแจ้งเตือน เช่น “ออเดอร์ของคุณพร้อมเสิร์ฟแล้ว กรุณามารับได้ที่เคาน์เตอร์” หรือ “พนักงานกำลังนำอาหารไปเสิร์ฟให้คุณ” เป็นต้น
- **แจ้งเตือนเครดิตหมดอายุ:** ตามที่กล่าวในระบบ Wallet ระบบจะส่งข้อความแจ้งเตือนผู้ใช้เป็นระยะก่อนที่เครดิตในกระเป๋าจะหมดอายุ (เช่น ก่อน 3 เดือน, 2 เดือน, 1 เดือน) เพื่อกระตุ้นการใช้งาน
- **ประกาศ/ข้อความประชาสัมพันธ์:** อาจมีการใช้ LINE OA ในการส่งข่าวสาร โปรโมชั่นใหม่ หรือประกาศปิดปรับปรุงสนามให้กับผู้ใช้ทุกคนหรือเป็นกลุ่ม (ฟีเจอร์นี้นอกเหนือจากระบบอัตโนมัติ อาจเป็นการ broadcast message ที่ผู้ดูแลส่งเอง)
- **หลักการส่งข้อความ:**
- เนื้อความต้องสั้น กระชับ และมีลิงก์หรือปุ่มที่นำผู้ใช้ไปยัง LIFF เพื่อดูรายละเอียดเพิ่มเติม
- หลีกเลี่ยงการส่งข้อมูล **ที่เป็นความลับในตัวข้อความโดยตรง** (เช่น เลข PIN, ข้อมูลส่วนตัว) เนื่องจากแชท LINE อาจไม่ปลอดภัยหรือผู้อื่นอาจเห็นหน้าจอ ควรใช้วิธีให้ผู้ใช้กดเพื่อเข้าสู่ระบบเพื่อดูข้อมูลเหล่านั้นแทน
- ควรจัดกลุ่มหรือจัดการ token ของผู้ใช้ให้พร้อมสำหรับส่ง (เช่น ใช้ UserID ของ LINE ในการ push) และจัดการกรณีผู้ใช้บล็อกหรือยกเลิกการติดตาม LINE OA ด้วย
- **การทดสอบและขอบเขต:** ควรทดสอบการส่งข้อความจริงบน LINE OA Sandbox/Production ให้ครบทุกกรณี เพื่อให้แน่ใจว่าผู้ใช้ได้รับข้อความถูกต้อง และปริมาณการส่งไม่เกินขีดจำกัดของ Messaging API ในแต่ละเดือน (หากฐานผู้ใช้เยอะต้องพิจารณาอัตราการส่ง)

7.7 ความปลอดภัยและระบบบันทึกการใช้งาน (Security & Traceability)

เพื่อความมั่นใจในเสถียรภาพของระบบและป้องกันปัญหาด้านความปลอดภัย ทีมพัฒนาจะต้องออกแบบให้ระบบมีมาตรการดังต่อไปนี้:

- **การจัดเก็บและสำรองข้อมูล:** ฐานข้อมูลที่ใช้เก็บข้อมูลการจอง ลูกค้า การทำธุรกรรมต่างๆ จะต้องมีความปลอดภัย มีการสำรองข้อมูล (Backup) เป็นระยะ และมีการควบคุมสิทธิ์การเข้าถึงข้อมูล (เช่น ใช้การเข้ารหัสข้อมูลที่อ่อนไหว หรือจัดเก็บ Token/Key ต่างๆ ในระบบที่ปลอดภัยเช่นใน Environment Variables หรือบริการ Secret Manager)
- **การจัดการข้อมูลบัตรเครดิต:** ระบบจะไม่จัดเก็บข้อมูลบัตรเครดิตของลูกค้าโดยตรงในระบบเรา แต่จะใช้วิธี tokenization ของ Omise (OPN) ทั้งหมด ดังนั้นความเสี่ยงด้านข้อมูลบัตรเครดิตรั่วไหลจะลดลง
- **ความปลอดภัยของรหัส PIN:** รหัส PIN สำหรับเข้าพื้นที่ที่ระบบสร้างให้แต่ละครั้งจะต้องถูกจัดเก็บหรือประมวลผลอย่างปลอดภัย:
- เมื่อจัดเก็บในฐานข้อมูล ควรจัดเก็บในรูปแบบ **Hash** (one-way hash) หรือเข้ารหัส ไม่เก็บเป็นตัวเลข 4-6 หลักตรงๆ
- การสร้าง/ขอดู PIN ในระบบจะต้องจำกัดสิทธิ์ (เฉพาะผู้ใช้ที่เกี่ยวข้องหรือ admin ที่จำเป็น) และมีการ **จำกัดจำนวนครั้ง** ในการร้องขอดู PIN เพื่อป้องกัน misuse
- PIN แต่ละรายการมี **อายุการใช้งานจำกัด** ชัดเจน ระบบจะทำงานร่วมกับ Igloohome เพื่อ **ยกเลิก PIN** ทันทีที่พ้นช่วงเวลาที่อนุญาต
- **Audit Trail (บันทึกการใช้งาน):** ระบบจะต้องเก็บ **Log กิจกรรม** ที่สำคัญทุกครั้ง เช่น การเข้าสู่ระบบของผู้ใช้หลังบ้าน, การสร้าง/แก้ไข/ลบข้อมูลสำคัญ (สถานที่ สนาม สินค้า โปรโมชั่น), การทำธุรกรรมการเงิน (เติมเงิน, ใช้เงิน, คืนเงิน) และการเรียกใช้งาน API กับบริการภายนอก (เช่น เรียกสร้าง PIN) โดย log ควรประกอบด้วยรายละเอียด เช่น ใครทำอะไร เมื่อไร จาก IP/device ไหน เพื่อใช้ในการตรวจสอบย้อนหลังได้
- **อัตราการรองรับโหลด (Performance):** ระบบควรได้รับการออกแบบให้รองรับ **การใช้งานพร้อมกันจำนวนมาก** ได้ในเกณฑ์ดี (High Concurrency) เป้าหมาย uptime ของระบบควรอยู่ที่ **99.9%** ขึ้นไป และการตอบสนองของระบบในฟังก์ชันสำคัญควรอยู่ในเกณฑ์ที่รวดเร็ว เช่น
- การค้นหาความพร้อมสนาม (availability lookup) ใช้เวลาไม่เกิน ~300 มิลลิวินาที
- การทำรายการ checkout ชำระเงิน ใช้เวลาไม่เกิน ~400 มิลลิวินาที (ไม่นับรวมเวลาที่ผู้ใช้กรอกข้อมูล)
- การสร้างและแสดงผล QR code ใบแจ้งหนี้ ใช้เวลาไม่เกิน ~200 มิลลิวินาที
(ค่าดังกล่าวเป็นเป้าหมายโดยประมาณ เพื่อให้แน่ใจว่าผู้ใช้ได้รับประสบการณ์ที่ดี)
- **การเฝ้าระวังและแจ้งเตือน (Monitoring & Alerts):** ระบบควรมีการทำงานร่วมกับเครื่องมือมอนิเตอร์ เช่น logging service หรือ APM ที่สามารถส่งการแจ้งเตือนเมื่อ:
- เซิร์ฟเวอร์ล่มหรือไม่ตอบสนอง
- Response time สูงผิดปกติ
- เกิด Error ในระบบจำนวนมาก
- เหตุการณ์สำคัญอื่นๆ (เช่น สต็อกเหลือน้อยกว่าที่กำหนด)
โดยทีมพัฒนาควรตั้งค่าการแจ้งเตือนไปยังผู้ที่เกี่ยวข้อง (เช่น DevOps ของทีม, ผู้ว่าจ้าง) เพื่อแก้ปัญหาได้อย่างทันท่วงที

8. โครงสร้างระบบและรายละเอียดเชิงเทคนิค

หัวข้อนี้จะอธิบายภาพรวมเชิงสถาปัตยกรรมของระบบ โดยแยกตามโมดูลและความรับผิดชอบ รวมถึงตัวอย่าง API และโมเดลข้อมูลคร่าวๆ เพื่อช่วยในการประเมินและออกแบบระบบ:

8.1 สถาปัตยกรรมโมดูล (Module Responsibilities & Interfaces)

ระบบถูกแบ่งออกเป็นส่วนๆ ตามหน้าที่เพื่อความชัดเจนในการพัฒนาและบำรุงรักษา แต่ละส่วนมีความรับผิดชอบดังนี้:

- **Client (LIFF PWA):** ส่วนของ Web App ที่รันบน LINE LIFF ซึ่งผู้ใช้ปลายทางใช้งาน ทำหน้าที่จัดการการแสดงผลและอินเทอร์แอคชัน เช่น การล็อกอินผ่าน LINE, เลือกสถานที่/สนาม, การทำรายการจองและสร้างตะกร้าสินค้า, การแสดงยอด Wallet และเติมเงิน, การทำ Checkout ชำระเงิน (ประสานงานกับ Payment Gateway), การแสดง PIN และใบเสร็จให้ผู้ใช้, รวมถึงฟังก์ชันการสแกน QR (สำหรับจ่ายเงินหรือเข้าถึงสิทธิ์ต่างๆ)
- **Admin Portal (Web Admin):** ส่วนเว็บสำหรับผู้ดูแลระบบ ทำหน้าที่จัดการข้อมูลและตั้งค่าต่างๆ ตามที่กล่าวในหัวข้อ 7.5 เช่น การกำหนดกฎการชำระเงิน, สร้าง/แก้ไขแพ็กเกจโปรโมชั่น, จัดการผู้ใช้งานและสิทธิ์, ดูและส่งออกรายงาน ฯลฯ
- **Authentication & Authorization Module:** จัดการการล็อกอินผ่าน LINE (Line Login OAuth2) รับ Token และตรวจสอบผู้ใช้, จัดเก็บโปรไฟล์ผู้ใช้เบื้องต้น (เช่น LINE ID, ชื่อ), กำหนดบทบาท (Role) ตามที่ได้รับมอบหมาย, และตรวจสอบสิทธิ์ (RBAC) เมื่อมีการเรียกใช้งาน API แต่ละส่วน ทั้งนี้รวมถึงการเก็บ **Audit Logs** การเข้าใช้งานของผู้ดูแล
- **Venue Management Module:** จัดการข้อมูลสถานที่และสนาม เช่น API สำหรับเพิ่ม/แก้ไข/ลบสถานที่และคอร์ท, จัดเก็บข้อมูลเวลาทำการ ช่วงปิด (blackout) และ Layout ของสนาม, และมีบริการสำหรับ **ตรวจสอบความว่าง (Availability Engine)** ของสนามเมื่อมีการค้นหา/จอง (จะประสานข้อมูลการจองที่มีอยู่กับเวลาเปิดปิดและ blackout)
- **Booking Module:** จัดการกระบวนการจองสนาม ตั้งแต่การสร้างการจองใหม่, การปรับแก้หรือยกเลิกการจอง (ถ้าอนุญาต), การคำนวณราคาตามทรัพยากรที่จองและบริการเสริม, การตรวจสอบสิทธิ์โปรโมชั่น, การล็อกสนามเมื่อจอง (กันไม่ให้คนอื่นจองซ้ำ), **การสร้าง PIN** เมื่อมีการยืนยันจอง, และการเชื่อมต่อกับระบบแจ้งเตือนเพื่อส่งรายละเอียดให้ลูกค้า
- **Wallet Module:** จัดการบัญชีกระเป๋าเงินของผู้ใช้ เก็บ **บัญชีแยก (Ledger)** ของเครดิตแต่ละคน (เพิ่มเมื่อเติมเงิน, ลดเมื่อใช้จ่าย, หมดอายุเมื่อครบกำหนด), มีตัวทำงาน **Scheduler สำหรับหมดอายุเครดิต** ที่จะคอยตรวจสอบเครดิตที่ครบ 12 เดือนและตัดทิ้ง พร้อมทั้งเรียกใช้ระบบแจ้งเตือนส่งข้อความแจ้งผู้ใช้, รองรับฟังก์ชัน **โอน/ให้ของขวัญ (Gift Credits)** ที่เป็นการ transfer เครดิตให้ผู้ใช้อื่นในระบบ
- **Payment & Checkout Module:** ประสานงานขั้นตอนการชำระเงินเมื่อผู้ใช้ทำการ Checkout:
- ตรวจสอบ **ความถูกต้อง** ของรายการ (เช่น ยอดเงิน, ช่องทางชำระ, โปรโมชั่นที่ใช้)
- **ตรวจสอบประเภทบัตร** (เช่น จากหมายเลขบัตรที่กรอกมาว่าเป็นบัตรต่างประเทศหรือไม่) เพื่อเพิ่มค่าธรรมเนียม 15 บาทถ้าจำเป็น
- สร้าง **Charge** ผ่าน Omise หรือเตรียมข้อมูล PromptPay QR ให้ผู้ใช้
- รับข้อมูล **webhook** จาก Omise แจ้งสถานะการชำระ (สำเร็จ/ล้มเหลว) แล้วแจ้งผลกลับไปยังระบบจองหรือ Order ให้ทราบ

- **Order Management / POS & KDS Module:** จัดการรายการสั่งซื้อสินค้าและอาหาร เครื่องดื่ม:
- รองรับการ **สร้างออเดอร์ (Order)** จากฝั่ง POS หรือการสั่งผ่าน LIFF, เก็บสถานะออเดอร์ (เช่น ใหม่, กำลังเตรียม, พร้อมเสิร์ฟ, สำเร็จ, ยกเลิก)
- ถ้ามาจาก POS และเลือกชำระด้วย Wallet, โมดูลนี้จะสร้าง **Invoice QR** พร้อมข้อมูลและจำนวนเงิน (payload) เพื่อให้ลูกค้าสแกนจ่ายใน LIFF หรือสร้างคำร้องสำหรับให้พนักงานสแกน Wallet QR ของลูกค้าได้
- เมื่อได้รับการยืนยันชำระ (เช่น ลูกค้าสแกนจ่ายแล้ว, ระบบ Wallet หักเงินสำเร็จ) ก็อัปเดตสถานะออเดอร์เป็นจ่ายแล้ว (**paid**) และส่งข้อมูลไปยัง KDS (Kitchen Display System) เพื่อขึ้นคิวทำอาหาร
- รองรับการ **เปลี่ยนสถานะ** ออเดอร์ตาม workflow (เช่น จาก paid -> preparing -> ready -> picked up) และให้พนักงานหน้าร้านอัปเดตได้
- รองรับการ **Refund/Void** ออเดอร์ในกรณีที่ลูกค้ายกเลิกหรือสินค้าขาด เป็นต้น โดยประสานกับ Wallet เพื่อคืนเครดิต (ถ้าจ่ายไปแล้ว) และอาจต้อง **คืนสต็อกสินค้า/วัตถุดิบ** หากของยังไม่ถูกใช้
- **Inventory/BOM Module:** จัดการสต็อกและสูตรส่วนประกอบ:
- เมื่อมีการสร้างออเดอร์ประเภทอาหาร/เครื่องดื่มและออเดอร์นั้นเข้าสู่สถานะ “กำลังเตรียม (Preparing)” ระบบจะทำการ **ตัดสต็อกวัตถุดิบอัตโนมัติ** ตาม BOM ของเมนูนั้นๆ
- หากสินค้าหรือวัตถุดิบรายการใด **เหลือน้อยกว่ากำหนด (Low Stock)** โมดูลนี้สามารถส่งการแจ้งเตือนไปยัง Admin/Venue Manager เพื่อเติมสต็อก
- ให้ API สำหรับอัปเดตสูตร BOM ของเมนู และดูระดับสต็อก (inventory levels) เพื่อใช้ในหน้าหลังบ้าน
- **Messaging/Notification Module:** รับผิดชอบการส่งข้อความแจ้งเตือนผ่าน LINE OA ตามเหตุการณ์ต่างๆ ที่เกิดขึ้นในระบบ:
- เช่น เมื่อการจองถูกสร้าง -> ส่งข้อความยืนยัน, PIN พร้อมลิงก์
- เมื่อมีการออก PIN -> ส่ง PIN (หรือจัดการให้ผู้ใช้เห็น PIN ใน LIFF)
- เมื่อผู้ใช้เติมเงินหรือใช้เงิน -> ส่งใบเสร็จยืนยัน
- เมื่อออเดอร์พร้อม -> ส่งแจ้งเตือนให้มารับของ เป็นต้น
- โมดูลนี้ต้องเชื่อมกับ LINE Messaging API และดูแลอัตราการส่งข้อความ (rate limit) ด้วย
- **Accounting (FlowAccount) Integration:** จัดการการแมปข้อมูลธุรกรรมของระบบเข้าสู่ระบบบัญชี:
- กำหนดรูปแบบ mapping ของข้อมูล เช่น การเติมเครดิต (เงินเข้ากระเป๋า) = บันทึกบัญชีเป็น **Deferred Revenue/Wallet Liability**, การใช้เครดิตซื้อของ = บันทึกเป็น **Revenue** ย้ายจาก liability มาที่รายได้
- จัดทำ **ไฟล์ส่งออก** หรือข้อมูล API ให้ตรงตามที่ FlowAccount ต้องการ เช่น รายการ Journal Entry หรือ Invoice/Receipt ที่ต้องลงบัญชี
- ทำ **การกระทบยอด (Reconcile)** หากจำเป็น เช่น ตรวจว่าเงินที่เก็บจาก Omise ตรงกับยอดขายที่บันทึกหรือไม่
- **Reporting Module:** รวมรวมข้อมูลจากส่วนต่างๆ เพื่อสร้าง **รายงาน**:
- คำนวณ P&L (รายได้-ค่าใช้จ่าย) ของแต่ละสาขารายเดือน และของบริษัทโดยรวม
- ติดตามยอดเงินใน Wallet ที่เป็นหนี้สินคงค้าง และเครดิตที่หมดอายุแล้ว

- จัดทำรายงานสรุปอื่นๆ และจัดเก็บ snapshot ตามช่วงเวลา (ReportSnapshot) เพื่อให้ admin เรียกดูย้อนหลังได้อย่างรวดเร็ว
- **Investor Portal (ถ้ามีใน Phase 1):** สำหรับให้นักลงทุนหรือเจ้าของดูข้อมูลสำคัญ:
- แสดง **สรุปผลประกอบการ (P&L)** รายเดือนหรือรายไตรมาส
- แสดง **ประวัติการจ่ายผลตอบแทน (Payout History)** หรือเงินปันผลที่โอนให้ผู้ลงทุน (หากมีระบบนั้น)
- ไม่มีฟังก์ชันอื่นที่เป็นการปฏิบัติการ (read-only อย่างเดียว)

แต่ละโมดูลข้างต้นจะสื่อสารกันผ่าน API ภายในและกับฐานข้อมูลส่วนกลาง รวมถึงการเรียก API ภายนอก (LINE, Omise, FlowAccount, Igloohome) ตามที่ระบุ ซึ่งทีมพัฒนาต้องออกแบบโครงสร้างระบบ (เช่น microservices หรือ modular monolith) ให้เหมาะสมเพื่อรองรับความสามารถทั้งหมดนี้

8.2 ตัวอย่าง API Interfaces

เพื่อแสดงภาพรวมของการออกแบบ API ที่เป็นไปได้สำหรับระบบ ต่อไปนี้คือตัวอย่าง Endpoint หลักที่อาจถูกพัฒนา:

```
GET    /venues/{venue_id}/availability        # ดึงข้อมูลความว่างของสนามในสถานที่ที่กำหนด (ตามวัน/เวลาที่ระบุผ่าน query param)
POST   /bookings                              # สร้างการจองใหม่ (ส่งรายละเอียดสถานที่ สนาม เวลา และบริการเสริมที่เลือก)
POST   /bookings/{booking_id}/pin/issue       # ออก PIN สำหรับการจองที่ระบุ (เรียกใช้ Igloohome API ภายใน)
PUT    /bookings/{booking_id}/cancel          # (ถ้ารองรับ) ยกเลิกการจอง

GET    /wallet                                # ดึงข้อมูลกระเป๋าเงินของผู้ใช้ (ยอดคงเหลือ, ประวัติธุรกรรมย่อๆ)
POST   /wallet/topup                          # สร้างคำขอเติมเงิน (เพื่อรับลิงก์/QR ชำระเงินจาก Omise)
POST   /wallet/gift                           # โอนเครดิต/ให้ของขวัญไปยังผู้ใช้รายอื่น
POST   /checkout/confirm                      # ดำเนินการตัดเงิน/ชำระเงินจาก Wallet (หรือช่องทางที่เลือก) เพื่อยืนยันการจอง/สั่งซื้อ

POST   /pos/orders                            # สร้างคำสั่งซื้อ (Order) ใหม่จาก POS หรือการสั่งใน LIFF
POST   /pos/orders/{order_id}/state           # อัปเดตสถานะการจัดเตรียมของ Order (เช่น เริ่มเตรียม เสร็จแล้ว)
POST   /pos/invoice                           # สร้าง Invoice QR สำหรับให้ลูกค้าสแกนจ่าย (ระบุ order_id และยอดเงิน)
POST   /wallet/pay-invoice                    # (สำหรับการสแกน wallet QR) หักเงินตาม invoice ที่สร้าง เมื่อได้รับอนุมัติจากผู้ใช้

GET    /reports/pnl?month=2025-09             # ดึงรายงาน P&L ของเดือนที่ระบุ (อาจแบ่งตามสาขา)
POST   /accounting/flowaccount/export         # เรียกการส่งออกข้อมูลบัญชีไปยัง FlowAccount (อาจระบุช่วงเวลาหรือประเภทข้อมูล)
```

รายการข้างต้นเป็นเพียงตัวอย่างเบื้องต้น ซึ่งรายละเอียดพารามิเตอร์และรูปแบบข้อมูลจริงจะถูกกำหนดในขั้นตอนการออกแบบ API โดยละเอียด แต่จากตัวอย่างจะเห็นภาพฟังก์ชันหลักที่ระบบต้องรองรับ

8.3 โมเดลข้อมูล (High-level Data Model)

เพื่อให้เห็นภาพรวมของข้อมูลที่จะต้องจัดเก็บในระบบ ต่อไปนี้คือโมเดลข้อมูลหลัก (Entity) และความสัมพันธ์ในระบบนี้:

- **User** – แทนข้อมูลผู้ใช้งานระบบ (ลูกค้า/แอดมิน): ประกอบด้วยรหัสผู้ใช้, LINE ID (ใช้เป็น unique key ในระบบ), ชื่อ, อีเมล (ถ้ามี), บทบาท (roles) เช่น Customer/Admin, และข้อมูลอื่นๆ เช่น วันที่สมัคร
- **Venue** – ข้อมูลสาขาหรือสถานที่: เช่น รหัสสาขา, ชื่อสถานที่, ตำแหน่งที่ตั้ง (ที่อยู่, พิกัด), เวลาเปิดปิด, รายละเอียดติดต่อ, เป็นต้น
- **Court** – ข้อมูลสนาม (คอร์ท) แต่ละสนามในสถานที่: เช่น รหัสสนาม, อ้างอิงถึง Venue ที่อยู่, ชื่อหรือหมายเลขสนาม, ประเภทสนาม, สถานะ (ว่าง/ปิดซ่อม)
- **Booking** – การจองสนาม: เช่น รหัสการจอง, อ้างอิงถึง Venue และ Court, อ้างอิงถึง User (ผู้จอง), เวลาเริ่มต้นและเวลาสิ้นสุดการจอง, จำนวนผู้เข้าร่วม, รายการบริการเสริมที่เลือก, ยอดชำระ, สถานะ (ยืนยันแล้ว/ยกเลิก)
- **Wallet** – กระเป๋าเงินของผู้ใช้: ผูกกับ User (1-ต่อ-1), ยอดคงเหลือปัจจุบัน, ประวัติหรือรายการเคลื่อนไหว (อาจเก็บในตาราง transaction แยก แต่สามารถมองว่า Wallet มี relation ไป transaction history ได้)
- **PaymentTransaction** – รายการธุรกรรมการเงิน: เช่น รหัสรายการ, ประเภท (เติมเงิน, ตัดเงิน, refund), วิธีการ (PromptPay, Visa domestic, Visa intl ฯลฯ), จำนวนเงิน, ค่าธรรมเนียม, สถานะ (สำเร็จ/ล้มเหลว), ผูกกับ User หรือ Order/Booking ที่เกี่ยวข้อง, timestamp
- **ReportSnapshot** – เก็บข้อมูลสรุปรายงานช่วงเวลาต่างๆ: เช่น รหัสสาขา (หรือ HQ), เดือนปีที่ทำรายงาน, ข้อมูล JSON สรุป P&L ของเดือนนั้น, วันที่สร้างรายงาน
- **AuditLog** – (ถ้ามีแยก) บันทึกเหตุการณ์สำคัญ: เช่น ไอดีเหตุการณ์, ประเภท (login, data_change, error), รายละเอียด, ผู้เกี่ยวข้อง, timestamp

โมเดลดังกล่าวเป็นเพียงภาพรวมระดับสูงเพื่อแสดงประเภทข้อมูลหลักๆ โดยในขั้นพัฒนาจริง อาจมีการเพิ่มตาราง/ฟิลด์รายละเอียดมากกว่านี้ (เช่น ตาราง Role/Permission, ตาราง Promotion, ตาราง Mapping ระหว่าง Booking กับ Add-on ที่เลือก เป็นต้น) ตามความจำเป็นของการใช้งาน

8.4 เหตุการณ์ระบบ (Domain Events)

ระบบนี้มีลักษณะการทำงานที่โมดูลต่างๆ ส่งผลกระทบซึ่งกันและกัน การออกแบบให้รองรับ **Event-Driven** จะช่วยให้แต่ละส่วนทำงานสอดคล้องกันได้อย่างถูกต้อง ตัวอย่าง **Domain Events** ที่ระบบควรจัดการ ได้แก่:

- **booking.created:** เมื่อมีการสร้างการจองสนามใหม่ -> ระบบ Booking จะ **สร้าง PIN** สำหรับการเข้าพื้นที่ และ **แจ้งเตือน LINE** ไปยังผู้จอง รวมถึง **บันทึกข้อมูลการจอง** ลงรายงานสถิติ (เช่น เพิ่ม count การใช้งานสนาม)
- **pin.issued:** เมื่อระบบได้ทำการออก PIN สำหรับการจองใดๆ -> ระบบ Notification จะส่งข้อความ LINE ไปยังผู้ใช้ที่เกี่ยวข้องเพื่อแจ้ง PIN (หรือให้เข้าดู PIN ในแอป)

- **wallet.topup.settled:** เมื่อมีการเติมเงินเข้า Wallet และได้รับการยืนยัน (Omise webhook) -> ระบบ Wallet จะเพิ่มเครดิตให้ผู้ใช้, และ **โมดูลบัญชี** จะบันทึกว่ามียอดเงินรับเข้าซึ่งถือเป็นหนี้สิน (liability) ใน FlowAccount
- **order.paid:** เมื่อคำสั่งซื้อสินค้า/อาหารได้รับการชำระเงินเรียบร้อย -> ระบบ KDS/Order จะทำการแจ้งไปยังครัวให้เริ่มเตรียมออเดอร์ และ **ระบบ Inventory** จะตรวจสอบวัตถุดิบที่ต้องใช้ (ถ้าคำสั่งมีอาหาร) เตรียมตัดสต็อกเมื่อเริ่มทำ
- **order.preparing:** เมื่อพนักงานหรือระบบเปลี่ยนออเดอร์เป็นสถานะ “กำลังเตรียม” -> **Inventory Module** จะทำการหักสต็อกวัตถุดิบตาม BOM ของเมนูที่สั่งทันที (ถือว่าเริ่มใช้วัตถุดิบ)
- **order.ready:** เมื่อครัวเตรียมอาหารเสร็จ (สถานะ ready) -> ระบบอาจแจ้งเตือนผู้ใช้ (ถ้าสั่งเอง) หรือแจ้งพนักงานเสิร์ฟ
- **order.voided/refunded:** เมื่อออเดอร์ถูกยกเลิกหรือคืนเงิน -> ระบบ Wallet จะได้รับ event เพื่อทำ **เครดิตเงินคืน** ให้ผู้ใช้ (หากมีการชำระเงินไปแล้ว) และอาจต้อง **คืนสต็อกสินค้า/วัตถุดิบ** หากของยังไม่ถูกใช้
- **inventory.low_stock:** เมื่อระดับสต็อกของรายการใดต่ำกว่าที่กำหนด -> ระบบ Inventory สร้าง event นี้ -> **Admin Portal** รับ event เพื่อแสดงแจ้งเตือนหรือส่ง LINE แจ้งผู้จัดการสาขาให้ทราบ (ถ้าตั้งค่าไว้)
- **accounting.export.ready:** เมื่อมีการสร้างไฟล์หรือตรวจสอบข้อมูลสำหรับส่งบัญชีเสร็จ -> ระบบอาจแจ้งเตือนผู้ดูแลว่าข้อมูลพร้อมแล้ว (หรือส่งอีเมลพร้อมไฟล์แนบ)

การใช้ Event ภายในลักษณะนี้จะช่วยให้การประสานงานระหว่างส่วนต่างๆ ในระบบเกิดขึ้นอย่างเป็นระบบ และง่ายต่อการบำรุงรักษา/เพิ่มฟังก์ชัน (เช่น ถ้าต้องการเพิ่มการแจ้งเตือน email ทุกครั้งที่เกิด event บางอย่าง ก็สามารถ subscribe event นั้นเพิ่มได้โดยไม่กระทบ logic หลัก)

9. การปรับใช้ระบบและแผนการเปิดตัว (Deployment & Rollout)

- **การติดตั้งระบบ (Deployment):** ทีมพัฒนาจะต้องเตรียมสภาพแวดล้อมการทำงานให้พร้อมสำหรับการรันระบบทั้งส่วน Front-end (LIFF PWA) และ Back-end (API/Database) โดยคำนึงถึงการ **รองรับผู้ใช้จำนวนมาก** และ **ความปลอดภัย** เช่น
- ใช้เซิร์ฟเวอร์หรือบริการคลาวด์ที่มีความเสถียร (อาจพิจารณา Cloud Functions, Container, หรือ VM ตามความเหมาะสม)
- แยกสภาพแวดล้อม **Development/Staging** สำหรับทดสอบ ออกจาก **Production** ที่ใช้งานจริง
- ตั้งค่า CDN หรือ caching สำหรับเนื้อหา static ของ PWA เพื่อเพิ่มความเร็ว และเปิดใช้ SSL (HTTPS) สำหรับโดเมนทุกครั้ง
- **การทดสอบก่อนเปิดจริง:** หลังพัฒนาสมบูรณ์ ทีมพัฒนาควรจัดให้มีช่วง **Soft Launch/UAT** ในสภาพแวดล้อม Staging หรือ Production (ในวงจำกัด) เพื่อทดสอบการทำงานกับผู้ใช้จริงจำนวนหนึ่ง เก็บ feedback และแก้ไขข้อบกพร่องก่อนเปิดให้ผู้ใช้ทั้งหมดใช้งาน
- **การฝึกอบรมพนักงาน:** ก่อนเปิดระบบจริง ควรมีการ **อบรมการใช้งาน** ให้กับพนักงานที่เกี่ยวข้อง เช่น วิธีใช้ระบบหลังบ้าน วิธีใช้หน้าจอ POS และ KDS, ขั้นตอนการช่วยเหลือลูกค้าเมื่อพบปัญหา เพื่อให้การเริ่มต้นใช้งานราบรื่น
- **การส่งมอบและรับมอบงาน:** เมื่อครบกำหนดระยะเวลาพัฒนาและทดสอบ ทีมพัฒนาจะทำการ **ส่งมอบระบบ** ให้ผู้ว่าจ้าง ตรวจสอบความครบถ้วนตาม TOR นี้ หากผ่านการ

ทดสอบยอมรับ (UAT) ก็จะถือว่าการพัฒนาในเฟส 1 เสร็จสิ้นสมบูรณ์ และเข้าสู่ช่วงรับประกัน/สนับสนุน 60 วันตามที่ระบุ

- **แผนเฟส 2 (อนาคต):** หลังจากระบบเฟส 1 เสถียรและใช้งานจริงแล้ว ผู้ว่าจ้างอาจพิจารณาดำเนินการพัฒนาเฟส 2 ต่อไป ซึ่งประกอบด้วยฟังก์ชันที่วางแผนไว้ (Analytics, Loyalty/Membership, Referral, DUPR integration, ฯลฯ) โดยการออกแบบระบบเฟส 1 ได้เตรียมรองรับไว้แล้ว และในการเสนอราคาหรือแผนงานอนาคต ทีมพัฒนาสามารถนำข้อมูลเชิงเทคนิคจากเฟส 1 นี้ไปประเมินการขยายผลต่อได้ง่ายยิ่งขึ้น

สรุป: เอกสาร TOR ฉบับนี้ได้กำหนดคุณลักษณะ ขอบเขตงาน และข้อกำหนดทางเทคนิคโดยละเอียดสำหรับการพัฒนาระบบจองสนามกีฬาและขายบริการบน LINE LIFF PWA (Phase 1) ไว้อย่างครบถ้วน ทีมผู้พัฒนาที่รับงานควรศึกษารายละเอียดทั้งหมดเพื่อประเมินปริมาณงาน ระยะเวลา และทรัพยากรที่ต้องใช้ในการดำเนินการให้ตรงตามที่กำหนด เมื่อส่งมอบงานแล้ว ระบบที่ได้จะต้องสามารถให้บริการแก่ผู้ใช้ตามที่อธิบายไว้ได้จริงอย่างมีประสิทธิภาพ และมีความพร้อมสำหรับการขยายฟีเจอร์เพิ่มเติมในอนาคตต่อไป
