ระบบฟิตเนสครบวงจร (Phaloyainong Fitness)

เสนอ

ดร.สุณิสา สถาพรวจนา

อ.กิตติพงศ์ วะระทรัพย์

จัดทำโดย

นายรัชพล กลิ่นประทุม 66130500074

รับผิดชอบในส่วน Project Planning & Logical Database Design (20%)

นายวิชญพลช กังวานพณิชย์ 66130500078

รับผิดชอบในส่วน DDL Script, Data Dictionary and Insert Statements (20%)

นายกันดีล หริมโต๊ะสัน 66130500090

รับผิดชอบในส่วน Logical Database Design & SQL Statements (20%)

นายธนวัฒน์ ช่วยนุกูล 66130500098

รับผิดชอบในส่วน Logical Database Design & SQL Statements (20%)

นายธนภัทร คำพิบูลย์ 66130500103

รับผิดชอบในส่วน Business Requirements & Reports (20%)

รายงานนี้เป็นส่วนหนึ่งรายวิชา Database Management System

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

Table of Contents

เรื่อง	หน้า
Table of Contents	1
Business Requirements	2
Logical DB Design (Table Design)	19
1. Table: User	19
2. Table: Admin	20
3. Table: Package	20
4. Table: User_Package	21
5. Table: Trainer	22
6. Table: Trainer_Schedule	23
7. Table: Trainer_Booking	24
8. Table: Gym_Access_Log	25
9. Table: Branch	26
10. Table: Payment	27
11. Table: Credit_History	28
12. Table: Topup_Transaction	28
13. Table: Rank	29
14. Table: User_Refer_Setting	29
15.Table: User_Refer_Transaction	30
16.Table: Trainer_Review	30
17.Table: Coupon	31
SQL Statements	32
SQL Statement 1	34
SQL Statement 2	33
SQL Statement 3	34
SQL Statement 4	35
SQL Statement 5	36
Data Dictionary	37
DDL Script for Creating the Database	51
Export Data in the format of INSERT statements	60

Business Requirements

1.1 การสมัครสมาชิกและการเข้าสู่ระบบProject Overview (ภาพรวมโครงการ)

ชื่อโครงการ : การสร้างฐานข้อมูลสำหรับ Fitness

ชื่อว่า Phaloyainong Fitness

วัตถุประสงค์ : Phaloyainong Fitness

คือระบบฟิตเนสครบวงจรที่ออกแบบมาเพื่อตอบโจทย์ทั้งผู้ใช้งานและผู้ดูแลระบบ โดยฝั่งผู้ใช้สามารถสมัครสมาชิก, เลือกแพ็คเกจฟิตเนส, จองเทรนเนอร์, ชำระเงิน, สะสมแต้ม และติดตามสถานะการใช้งาน รวมถึงคำนวณค่าใช้จ่ายต่างๆได้อย่างสะดวก ขณะที่ฝั่งผู้ดูแลระบบสามารถจัดการข้อมูลพื้นฐาน, ตั้งค่าระบบแนะนำเพื่อน, จัดการแพ็คเกจและคูปอง ตลอดจนดูรายงานต่างๆ เพื่อวิเคราะห์ข้อมูลและพัฒนาการให้บริการ ระบบนี้ช่วยเพิ่มประสิทธิภาพ, ความสะดวก, และความโปร่งใส พร้อมทั้งส่งเสริมการตลาดและการตัดสินใจทางธุรกิจทำให้ PHALO เป็นทางเลือกที่ครบวงจรสำหรับธุรกิจฟิตเนส

Scope (ขอบเขตงาน)

ฟีเจอร์ สำหรับระบบแอปพลิเคชันฟิตเนสและระบบหลังบ้านมีดังนี้

1. ระบบการจัดการผู้ใช้ (User Management)

- ผู้ใช้สามารถสมัครสมาชิกด้วยการระบุชื่อผู้ใช้ (username), รหัสผ่าน, อีเมล, และเบอร์โทรศัพท์
- ระบบต้องตรวจสอบความเป็นเอกลักษณ์ของชื่อผู้ใช้และอีเมล (ไม่ซ้ำกับในระบบ)
- ผู้ใช้สามารถเข้าสู่ระบบด้วยชื่อผู้ใช้หรืออีเมลและรหัสผ่าน

1.2 โปรไฟล์ผู้ใช้

- ผู้ใช้สามารถแก้ไขข้อมูลส่วนตัว เช่น ชื่อ, ที่อยู่, เบอร์โทรศัพท์, และรหัสผ่าน
- ผู้ใช้สามารถดูเครดิตที่มีอยู่และประวัติการใช้งานเครดิต (Credit_History)
- ระบบต้องแสดงระดับยศ (Rank) และคะแนนสะสม (rank_point) ของผู้ใช้

1.3 การจัดการเครดิตและการเติมเงิน

- ผู้ใช้สามารถเติมเครดิตผ่านระบบ (Topup_Transaction)
- ระบบต้องบันทึกประวัติการเติมเงินและแสดงสถานะการทำธุรกรรม
- ผู้ใช้สามารถดูประวัติการเคลื่อนไหวของเครดิต (Credit_History)

  รวมถึงยอดคงเหลือก่อนหน้าและหลังการทำธุรกรรม

1.4 ระบบแนะนำเพื่อน (Referral System)

- การตั้งค่าโดยผู้ดูแลระบบ:

  - ผู้ดูแลระบบ (Admin)

    สามารถตั้งค่าอัตราส่วนเครดิตและเงื่อนไขสำหรับการแนะนำเพื่อนในแต่ละระดับ

    (User_Refer_Setting)

  - อัตราส่วนเครดิตสำหรับแต่ละระดับ (level_1 ถึง level_4)

    จะถูกกำหนดโดยผู้ดูแลระบบและสามารถปรับเปลี่ยนได้ตามนโยบายของบริษัท

- การแนะนำเพื่อนโดยผู้ใช้:

  - ผู้ใช้สามารถแนะนำเพื่อนให้มาสมัครใช้งานแอปพลิเคชันผ่านรหัสหรือลิงก์แนะนำเฉพาะของตน

  - เมื่อเพื่อนที่ถูกแนะนำทำการสมัครและทำธุรกรรมที่กำหนด ผู้ใช้ที่แนะนำจะได้รับเครดิตตามระดับและอัตราส่วนที่ตั้งไว้

  - ระบบรองรับการแนะนำเพื่อนถึง 4 ระดับ ซึ่งหมายความว่าผู้ใช้สามารถได้รับเครดิตจากการแนะนำของเพื่อนของเพื่อนถึง 4 ชั้น

- การป้องกันการใช้งานผิดวัตถุประสงค์:

  - เพื่อป้องกันการสะสมเครดิตอย่างไม่เหมาะสม ระบบจะมีการตรวจสอบและกำหนดเงื่อนไขต่าง ๆ เช่น การจำกัดจำนวนการแนะนำต่อผู้ใช้ หรือการตรวจสอบความถูกต้องของธุรกรรม

- User_Refer_Transaction จะบันทึกรายการเครดิตที่ผู้ใช้ได้รับจากการแนะนำเพื่อนในแต่ละระดับ
- ผู้ดูแลระบบสามารถปรับเปลี่ยนอัตราส่วนเครดิตและเงื่อนไขการแนะนำเพื่อนได้ตามความเหมาะสม

ตัวอย่างการทำงานของระบบแนะนำเพื่อน 4 ระดับ:

- ระดับที่ 1:

  - ผู้ใช้ A แนะนำ ผู้ใช้ B ให้มาสมัครใช้งานผ่านรหัสแนะนำของตน

  - เมื่อ ผู้ใช้ B ทำการซื้อแพ็คเกจหรือทำธุรกรรมที่กำหนด ผู้ใช้ A จะได้รับเครดิตตามอัตราส่วนที่ตั้งไว้สำหรับระดับที่ 1 (เช่น ได้รับ 10% ของยอดธุรกรรม)

- ระดับที่ 2:
    - ผู้ใช้ B แนะนำ ผู้ใช้ C ให้มาสมัครใช้งาน
    - เมื่อ ผู้ใช้ C ทำธุรกรรม ผู้ใช้ A จะได้รับเครดิตตามอัตราส่วนที่ตั้งไว้สำหรับระดับที่ 2 (เช่น ได้รับ 5% ของยอดธุรกรรมของ ผู้ใช้ C)
    - ผู้ใช้ B ก็จะได้รับเครดิตตามอัตราส่วนระดับที่ 1
- ระดับที่ 3:
    - ผู้ใช้ C แนะนำ ผู้ใช้ D ให้มาสมัครใช้งาน
    - เมื่อ ผู้ใช้ D ทำธุรกรรม ผู้ใช้ A จะได้รับเครดิตตามอัตราส่วนที่ตั้งไว้สำหรับระดับที่ 3 (เช่น ได้รับ 2% ของยอดธุรกรรมของ ผู้ใช้ D)
    - ผู้ใช้ B จะได้รับเครดิตระดับที่ 2 และ ผู้ใช้ C จะได้รับเครดิตระดับที่ 1
- ระดับที่ 4:
    - ผู้ใช้ D แนะนำ ผู้ใช้ E ให้มาสมัครใช้งาน
    - เมื่อ ผู้ใช้ E ทำธุรกรรม ผู้ใช้ A จะได้รับเครดิตตามอัตราส่วนที่ตั้งไว้สำหรับระดับที่ 4 (เช่น ได้รับ 1% ของยอดธุรกรรมของ ผู้ใช้ E)
    - ผู้ใช้ B จะได้รับเครดิตระดับที่ 3, ผู้ใช้ C ระดับที่ 2, และ ผู้ใช้ D ระดับที่ 1

2. ระบบการจัดการแพ็คเกจ (Package Management)

2.1 แพ็คเกจฟิตเนส

- ผู้ดูแลระบบสามารถสร้างและจัดการแพ็คเกจ (Package) โดยระบุชื่อ, คำอธิบาย, ราคา, ระยะเวลาใช้งาน, และคะแนนสะสมที่ได้รับ
- แพ็คเกจสามารถถูกเปิดหรือปิดการใช้งานตามความต้องการ (is_active)แพ็คเกจสร้างมาเพื่อให้ลูกค้ามีสิทธิ์เข้าไปใช้งานยิมได้

2.2 การซื้อแพ็คเกจโดยผู้ใช้

- ผู้ใช้สามารถเลือกซื้อแพ็คเกจที่ต้องการ (User_Package)
- ระบบต้องประมวลผลการชำระเงินสำหรับแพ็คเกจ (Payment)
- หลังจากการซื้อสำเร็จ ระบบต้องระบุวันเริ่มต้นและวันหมดอายุของแพ็คเกจ

3. ระบบการจองเทรนเนอร์ (Trainer Booking System)

3.1 ข้อมูลเทรนเนอร์

- ระบบต้องแสดงรายชื่อเทรนเนอร์ (Trainer) พร้อมรายละเอียด เช่น ชื่อ, ความเชี่ยวชาญ, ค่าบริการต่อชั่วโมง, และสถานะการทำงาน
- เทรนเนอร์มีตารางเวลาการทำงาน (Trainer_Schedule) ที่ระบุวันที่และเวลาที่พร้อมให้บริการ

3.2 การจองเทรนเนอร์

- ผู้ใช้สามารถจองเทรนเนอร์ตามตารางเวลาที่ว่าง (Trainer_Booking)
- ระบบต้องคำนวณค่าบริการตามจำนวนชั่วโมงและราคาต่อชั่วโมงของเทรนเนอร์
- ผู้ใช้ต้องทำการชำระเงินสำหรับการจองเทรนเนอร์ (Payment)

- หากในกรณีที่ลูกค้าไม่ได้สมัคร Package และมีการจองเทรนเนอร์ ให้ถือว่าลูกค้าคนนั้นมีสิทธิ์ใช้งานยิมได้ใช้ช่วงเวลานัด

3.3 การยกเลิกและคืนเงิน

- ผู้ใช้สามารถยกเลิกการจองได้ตามเงื่อนไขที่กำหนด
- ระบบต้องบันทึกเหตุผลการยกเลิกและปรับสถานะการจอง

3.4 ระบบรีวิวเทรนเนอร์

- ผู้ใช้สามารถให้คะแนนและเขียนรีวิวเทรนเนอร์หลังจากใช้บริการ (Trainer_Review)
- รีวิวและคะแนนจะถูกแสดงให้ผู้ใช้อื่น ๆ เห็นเพื่อประกอบการตัดสินใจ

4. ระบบการชำระเงิน (Payment System)

4.1 วิธีการชำระเงิน

- ระบบรองรับวิธีการชำระเงินหลากหลาย เช่น บัตรเครดิต, เดบิต, และเครดิตภายในระบบ
- ผู้ใช้สามารถใช้คูปองส่วนลดในการชำระเงิน (Coupon)

4.2 การประมวลผลการชำระเงิน

- ระบบต้องบันทึกรายละเอียดการชำระเงิน เช่น วันที่, จำนวนเงิน, ส่วนลด, และสถานะการชำระเงิน
- การชำระเงินต้องมีความปลอดภัยและเชื่อถือได้

4.3 การใช้คูปองและส่วนลด

- ผู้ใช้สามารถนำคูปองมาใช้ลดราคาในการชำระเงิน
- ระบบต้องตรวจสอบความถูกต้องและเงื่อนไขการใช้งานของคูปอง

5. ระบบคูปองและโปรโมชั่น (Coupon and Promotion System)

5.1 การสร้างและจัดการคูปอง

- ผู้ดูแลระบบสามารถสร้างคูปอง (Coupon) โดยระบุรหัสคูปอง, ชื่อ, คำอธิบาย, ส่วนลด, วันที่เริ่มและสิ้นสุด
- คูปองสามารถถูกกำหนดให้ใช้ได้กับผู้ใช้ที่มีระดับยศ (Rank) ที่กำหนด

5.2 การใช้งานคูปองโดยผู้ใช้

- ผู้ใช้สามารถใส่รหัสคูปองในขั้นตอนการชำระเงิน
- ระบบต้องตรวจสอบความถูกต้องของคูปองและปรับส่วนลดตามที่กำหนด

6. ระบบการจัดการสาขา (Branch Management)

6.1 ข้อมูลสาขา

- ระบบต้องเก็บข้อมูลสาขา (Branch) เช่น ชื่อ, ที่อยู่, เบอร์โทรศัพท์, อีเมล, เวลาเปิด-ปิด, และสถานะการเปิดใช้งาน
- ผู้ใช้สามารถเลือกสาขาที่ต้องการใช้บริการหรือจองเทรนเนอร์

6.2 การบันทึกการเข้าใช้บริการยิม

- เมื่อผู้ใช้เข้าใช้บริการยิม ระบบต้องบันทึกเวลาเข้าและออก (Gym_Access_Log)
- ระบบสามารถตรวจสอบสถานะการใช้งานแพ็คเกจของผู้ใช้เพื่อยืนยันสิทธิ์ในการเข้าใช้บริการ

7. ระบบการจัดการผู้ดูแลระบบ (Admin Management)

7.1 การจัดการผู้ดูแลระบบ

- ผู้ดูแลระบบสามารถเข้าสู่ระบบด้วยชื่อผู้ใช้และรหัสผ่าน
- ระบบต้องรองรับบทบาท (role) ของผู้ดูแลระบบที่แตกต่างกัน

7.2 การจัดการแพ็คเกจและคูปอง

- ผู้ดูแลระบบสามารถสร้าง, แก้ไข, และลบแพ็คเกจและคูปอง
- การเปลี่ยนแปลงต้องถูกบันทึกและระบุผู้ที่ทำการแก้ไข

7.3 การจัดการเทรนเนอร์

- ผู้ดูแลระบบสามารถเพิ่มและแก้ไขข้อมูลเทรนเนอร์

  รวมถึงสถานะการทำงานและการกำหนดสาขาที่ปฏิบัติงาน

7.4 การจัดการการตั้งค่าแนะนำเพื่อน

- ผู้ดูแลระบบสามารถกำหนดอัตราส่วนเครดิตที่ผู้ใช้จะได้รับจากการแนะนำเพื่อนในแต่ละระดับ

8. ระบบระดับยศและคะแนนสะสม (Rank and Point System)

8.1 การกำหนดระดับยศ

- ระบบมีระดับยศ (Rank) ที่กำหนดตามคะแนนสะสมของผู้ใช้
- แต่ละระดับยศมีเงื่อนไขคะแนนขั้นต่ำและอาจมีสิทธิพิเศษต่าง ๆ

8.2 การสะสมคะแนน

- ผู้ใช้จะได้รับคะแนนสะสม (rank_point) จากการซื้อแพ็คเกจและกิจกรรมต่าง ๆ
- คะแนนสะสมจะถูกใช้ในการกำหนดระดับยศและอาจแลกสิทธิพิเศษหรือส่วนลด

9. ระบบการจัดการตารางเวลาของเทรนเนอร์ (Trainer Schedule Management)

9.1 การสร้างตารางเวลาของเทรนเนอร์

- เทรนเนอร์สามารถระบุวันที่และเวลาที่พร้อมให้บริการ (Trainer_Schedule)
- ผู้ดูแลระบบสามารถช่วยจัดการและแก้ไขตารางเวลาของเทรนเนอร์

9.2 การตรวจสอบความขัดแย้งของตารางเวลา

- ระบบต้องตรวจสอบการจองเพื่อป้องกันการซ้อนทับของตารางเวลา
- เมื่อมีการจอง ระบบต้องอัปเดตสถานะของตารางเวลานั้น ๆ

Data Model (แบบจำลองข้อมูล)

ฐานข้อมูลประกอบด้วย **Entity** ดังนี้:

- **User**: ข้อมูลผู้ใช้ เช่น ชื่อ อีเมล และสถานะ
- **Admin**: ข้อมูลแอดมิน เช่น ชื่อและสิทธิ์
- **Package**: ข้อมูลแพ็กเกจ เช่น ชื่อและราคา
- **User_Package**: ข้อมูลการซื้อแพ็กเกจของผู้ใช้
- **Trainer**: ข้อมูลเทรนเนอร์ เช่น ความเชี่ยวชาญ
- **Trainer_Schedule**: ตารางเวลาของเทรนเนอร์
- **Trainer_Booking**: การจองเทรนเนอร์ของผู้ใช้
- **Trainer_Review**: รีวิวเทรนเนอร์จากผู้ใช้
- **Gym_Access_Log**: ประวัติการเข้า-ออกของผู้ใช้
- **Branch**: ข้อมูลสาขา เช่น ชื่อและที่อยู่
- **Payment**: ข้อมูลการชำระเงิน
- **Credit_History**: ประวัติการใช้เครดิต
- **Topup_Transaction**: ข้อมูลการเติมเงิน
- **Rank**: ระดับสมาชิก เช่น Silver, Gold
- **User_Refer_Setting**: การตั้งค่าระบบแนะนำเพื่อน
- **User_Refer_Transaction**: รายการแนะนำเพื่อน
- **Coupon**: ข้อมูลคูปองส่วนลด

ความสัมพันธ์ของแต่ละตาราง (Relationship) มีดังนี้:

Admin (ผู้ดูแลระบบ)

- ผู้ดูแลระบบ 1 คนสามารถสร้าง package ได้หลาย package
- ผู้ดูแลระบบ 1 คนสามารถสร้างคูปองได้หลายคูปอง

Branch (สาขา)

- 1 สาขาสามารถมีเทรนเนอร์ได้หลายคน
- 1 สาขาสามารถมีประวัติการเข้าใช้งาน gym ได้หลายรายการ
- 1 สาขาสามารถมีการจองเทรนเนอร์ได้หลายรายการ

Coupon (คูปอง)

- 1 คูปองสามารถถูกใช้ในการชำระเงินได้หลายครั้ง
- 1 คูปองถูกสร้างโดยผู้ดูแลระบบ 1 คน
- 1 คูปองสามารถผูกกับ rank ได้ 1 rank

Credit History (ประวัติการใช้เครดิต)

- 1 ประวัติการใช้เครดิตจะผูกกับผู้ใช้งาน 1 คน
- ประวัติการใช้เครดิตจะเก็บข้อมูลยอดก่อนและหลังการทำรายการ

Gym Access Log (ประวัติการเข้าใช้งาน)

- 1 ประวัติการเข้าใช้งานจะผูกกับผู้ใช้งาน 1 คน
- 1 ประวัติการเข้าใช้งานจะผูกกับสาขา 1 สาขา
- 1 ประวัติการเข้าใช้งานจะผูกกับ user_package 1 รายการ

Package (แพ็คเกจ)

- 1 package สามารถถูกซื้อโดยผู้ใช้งานได้หลายคน (ผ่าน user_package)
- 1 package สามารถให้ rank point ตามที่กำหนด
- 1 package ถูกสร้างโดยผู้ดูแลระบบ 1 คน

Payment (การชำระเงิน)

- 1 การชำระเงินสามารถใช้คูปองได้ 1 รายการ
- 1 การชำระเงินสามารถผูกกับ user_package ได้ 1 รายการ
- 1 การชำระเงินสามารถผูกกับการจองเทรนเนอร์ได้ 1 รายการ
- 1 การชำระเงินสามารถมีธุรกรรมการแนะนำ (user_refer_transaction) ได้หลายรายการ

Rank (ระดับสมาชิก)

- 1 rank สามารถมีผู้ใช้งานได้หลายคน
- 1 rank สามารถมีคูปองได้หลายรายการ

Topup Transaction (ธุรกรรมการเติมเงิน)

- 1 ธุรกรรมการเติมเงินจะผูกกับผู้ใช้งาน 1 คน

Trainer (เทรนเนอร์)

- 1 เทรนเนอร์สังกัดได้ 1 สาขา
- 1 เทรนเนอร์สามารถมีตารางงาน (trainer_schedule) ได้หลายรายการ
- 1 เทรนเนอร์สามารถมีการจอง (trainer_booking) ได้หลายรายการ
- 1 เทรนเนอร์สามารถมีรีวิว (trainer_review) ได้หลายรายการ

Trainer Booking (การจองเทรนเนอร์)

- 1 การจองเทรนเนอร์จะผูกกับผู้ใช้งาน 1 คน
- 1 การจองเทรนเนอร์จะผูกกับเทรนเนอร์ 1 คน
- 1 การจองเทรนเนอร์จะผูกกับสาขา 1 สาขา
- 1 การจองเทรนเนอร์จะผูกกับการชำระเงิน 1 รายการ

Trainer Review (รีวิวเทรนเนอร์)

- 1 รีวิวเทรนเนอร์จะผูกกับผู้ใช้งาน 1 คน
- 1 รีวิวเทรนเนอร์จะผูกกับเทรนเนอร์ 1 คน

Trainer Schedule (ตารางงานเทรนเนอร์)

- 1 ตารางงานจะผูกกับเทรนเนอร์ 1 คน

User (ผู้ใช้งาน)

- ผู้ใช้งาน 1 คนสามารถมี package ได้หลาย package (ผ่าน user_package)
- ผู้ใช้งาน 1 คนสามารถมีประวัติการใช้เครดิตได้หลายรายการ
- ผู้ใช้งาน 1 คนสามารถมีประวัติการเข้าใช้ gym ได้หลายครั้ง
- ผู้ใช้งาน 1 คนสามารถจองเทรนเนอร์ได้หลายครั้ง
- ผู้ใช้งาน 1 คนสามารถรีวิวเทรนเนอร์ได้หลายครั้ง
- ผู้ใช้งาน 1 คนสามารถมี rank ได้ 1 rank
- ผู้ใช้งาน 1 คนสามารถมีการตั้งค่าการแนะนำ (user_refer_setting) ได้ 1 รายการ
- ผู้ใช้งาน 1 คนสามารถถูกอ้างอิง (refer) โดยผู้ใช้งานคนอื่นได้หลายคน

User Package (แพ็คเกจของผู้ใช้งาน)

- 1 user_package จะผูกกับผู้ใช้งาน 1 คน
- 1 user_package จะผูกกับ package 1 รายการ
- 1 user_package จะผูกกับการชำระเงิน 1 รายการ
- 1 user_package สามารถมีประวัติการเข้าใช้งาน gym ได้หลายครั้ง

User Refer Setting (การตั้งค่าการแนะนำ)

- 1 การตั้งค่าการแนะนำจะผูกกับผู้ใช้งาน 1 คน
- การตั้งค่าการแนะนำกำหนดเปอร์เซ็นต์ที่จะได้รับสำหรับแต่ละระดับการแนะนำ
- 1 การตั้งค่าสามารถถูกแก้ได้โดยแอดมิน

User Refer Transaction (ธุรกรรมการแนะนำ)

- 1 ธุรกรรมการแนะนำจะผูกกับผู้ใช้งานที่แนะนำ 1 คน
- 1 ธุรกรรมการแนะนำจะผูกกับผู้ใช้งานที่ถูกแนะนำ 1 คน
- 1 ธุรกรรมการแนะนำจะผูกกับการชำระเงิน 1 รายการ

ฟีเจอร์ของระบบฐานข้อมูล **(Feauture)** ดังนี้:

ระบบผู้ใช้งาน (Users)

- จัดเก็บข้อมูลผู้ใช้งานพื้นฐาน (username, name, email, phone, address)
- ระบบเครดิต (credit) สำหรับใช้จ่ายภายในฟิตเนส
- ระบบแรงค์ (rank) มีการเก็บ rank point และระดับของสมาชิก
- ระบบแนะนำสมาชิก (user refer)
- บันทึกประวัติการใช้เครดิต (credit history)
- บันทึกประวัติการเข้าใช้งานฟิตเนส (gym access log)

ระบบสาขา (Branches)

- จัดการข้อมูลสาขาพื้นฐาน (name, phone, email, address)
- กำหนดเวลาเปิด-ปิด (opening_time, closing_time)
- ติดตามสถานะการให้บริการของสาขา (status)

ระบบแพ็คเกจ (Packages)

- จัดการข้อมูลแพ็คเกจ (name, description, price)
- กำหนดระยะเวลาของแพ็คเกจ (duration_days)
- กำหนดแรงค์พอยต์ที่จะได้รับ (earn_rank_point)
- ติดตามสถานะของแพ็คเกจ (status, is_active)

ระบบคูปอง (Coupons)

- จัดการข้อมูลคูปอง (code, name, description)
- กำหนดส่วนลด (discount) และระยะเวลาการใช้งาน
- ผูกกับระดับแรงค์ของผู้ใช้งาน
- ติดตามสถานะของคูปอง (status, is_active)

ระบบเทรนเนอร์ (Trainers)

- จัดเก็บข้อมูลเทรนเนอร์ (name, description, specialization)
- กำหนดเงินเดือนพื้นฐาน (base_salary) และค่าบริการต่อชั่วโมง (price_per_hour)
- จัดการตารางงาน (trainer schedule) กำหนดวันและเวลาทำงาน
- ระบบจองเทรนเนอร์ (trainer booking)
- ระบบรีวิวเทรนเนอร์ (trainer review) เก็บ rating และ comment
- ติดตามสถานะการให้บริการของเทรนเนอร์

ระบบการชำระเงิน (Payments)

- รองรับหลายช่องทางการชำระเงิน (payment_method)
- รองรับการใช้คูปองส่วนลด
- บันทึกข้อมูลการชำระเงิน (payment_date, amount, discount, total_amount)
- ติดตามสถานะการชำระเงิน (payment_status)

ระบบเติมเงิน (Topup)

- บันทึกธุรกรรมการเติมเงิน (amount)
- เก็บข้อมูลอ้างอิงการชำระเงิน (payment_reference)
- ติดตามสถานะการเติมเงิน (status)

ระบบแนะนำสมาชิก (User Refer)

- ตั้งค่าเปอร์เซ็นต์ผลตอบแทนแต่ละระดับ (level 1-4)
- บันทึกธุรกรรมการได้รับเครดิตจากการแนะนำ
- ติดตามระดับการแนะนำ (refer_level)

ระบบผู้ดูแล (Admin)

- จัดการข้อมูลผู้ดูแลระบบ (username, name, email)
- กำหนดบทบาทและสิทธิ์การใช้งาน (role)
- บันทึกการเข้าสู่ระบบ (last_login)
- ติดตามสถานะการใช้งาน (is_active)
- ตั้งค่า (user_refer_setting) เปอร์เซ็นต์ผลตอบแทนแต่ละระดับ

Logical DB Design (Table Design)

1. Table: User

- Columns:

  int user_id PK

  string username UK

  string password

  decimal credit

  string name

  string email UK

  string phone

  string address

  int rank_point

  int user_refer FK

  int rank_id FK

  datetime created_at

  datetime updated_at

  boolean is_active

- Relationships:

Many-to-One: user.rank_id ↔ rank.rank_id

Many-to-One: user.user_refer ↔ user.user_id

One-to-Many: user.user_id ↔ credit_history.user_id One-to-Many: user.user_id ↔ gym_access_log.user_id One-to-Many: user.user_id ↔ trainer_booking.user_id One-to-Many: user.user_id ↔ trainer_review.trainer_id One-to-Many: user.user_id ↔ topup_transaction.user_id One-to-Many: user.user_id ↔ user_package.user_id One-to-One: user.user_id ↔ user_refer_setting.user_id

2.Table: Admin

- Columns:

int admin_id PK

string username UK

string password

string name

string email UK

string role

datetime last_login

boolean is_active

3.Table: Package

- Columns:

int package_id PK

string name

string description

decimal price

int duration_days

int earn_rank_point

boolean is_active

int created_by FK

datetime created_at

datetime updated_at

- Relationship:

Many-to-One: package.created_by ↔ admin.admin_id ○ One-to-Many: package.package_id ↔ user_package.package_id

4.Table: User_Package

- Columns:

  int id PK

  int user_id FK

  int package_id FK

  int payment_id FK

  datetime start_date

  datetime exp_date

  string status

  int earn_rank_point

- Relationship:

Many-to-One: user_package.user_id ↔ user.user_id Many-to-One: user_package.package_id ↔ package.package_id Many-to-One: user_package.payment_id ↔ payment.payment_id

One-to-Many: user_package.id ↔ gym_access_log.user_package_id

5.Table: Trainer

- Columns:

int trainer_id PK

string name

string description

string specialization

decimal base_salary

decimal price_per_hour

string status

int branch_id FK

string address

datetime joined_date

boolean is_active

- Relationship:

Many-to-One: trainer_schedule.trainer_id ↔ trainer.trainer_id

6.Table: Trainer_Schedule

- Columns:

  int schedule_id PK

  int trainer_id FK

  date work_date

  time start_time

  time end_time

  string status

- Relationship:

Many-to-One: trainer_schedule.trainer_id ↔ trainer.trainer_id

7.Table: Trainer_Booking

- Columns

int booking_id PK

int user_id FK

int trainer_id FK

int branch_id FK

datetime booking_date

time start_time

time end_time

int hours

decimal price_per_hour

decimal total_amount

string status

string cancellation_reason

int payment_id FK

datetime created_at

- Relationship:

Many-to-One: trainer_booking.user_id ↔ user.user_id Many-to-One: trainer_booking.trainer_id ↔ trainer.trainer_id Many-to-One: trainer_booking.branch_id ↔ branch.branch_id Many-to-One: trainer_booking.payment_id ↔ payment.payment_id

8.Table: Gym_Access_Log

- Columns

int log_id PK

int user_id FK

int branch_id FK

int user_package_id FK

datetime check_in_time

datetime check_out_time

string status

- Relationship:

Many-to-One: gym_access_log.user_id ↔ user.user_id Many-to-One: gym_access_log.branch_id ↔ branch.branch_id Many-to-One: gym_access_log.user_package_id ↔ user_package.id

9.Table: Branch

- Columns:

  int branch_id PK

  string name

  string phone

  string email

  string address

  time opening_time

  time closing_time

  boolean is_active

- Relationships:

One-to-Many: branch.branch_id ↔ trainer.branch_id

One-to-Many: branch.branch_id ↔gym_access_log.branch_id

One-to-Many: branch.branch_id ↔ trainer_booking.branch_id

10.Columns: Payment

- Columns:

  int payment_id PK

  string payment_method

  datetime payment_date

  string payment_status

  string payment_note

  decimal amount

  decimal discount

  decimal total_amount

  string coupon_code FK

  datetime created_at

- Relationship:

One-to-Many: payment.payment_id ↔ user_package.payment_id

One-to-Many: payment.payment_id ↔ trainer_booking.payment_id

One-to-Many: payment.payment_id ↔ user_refer_transaction.payment_id

11.Table: credit_history

- Columns:

  int history_id PK

  int user_id FK

  decimal amount

  string type

  int reference_id

  decimal balance_before

  decimal balance_after

  datetime created_at

- Relationship:

Many-to-One: credit_history.user_id ↔ user.user_id

12.Table: Topup_Transaction

- Columns:

  int trans_id PK

  int user_id FK

  string status

  decimal amount

  string payment_reference

  datetime created_at

- Relationship:

Many-to-One: topup_transaction.user_id ↔ user.user_id

13.Table: Rank

- Columns:

  int rank_id PK

  string name

  int min_point

  string description

Relationship:

One-to-Many: rank.rank_id ↔ user.rank_id

One-to-Many: rank.rank_id ↔ coupon.rank_id

14.Table: User_Refer_Setting

- Columns:

  int user_id PK

  decimal level_1

  decimal level_2

  decimal level_3

  decimal level_4

  datetime created_at

  int updated_by FK

- Relationship:

One-to-One: user_refer_setting.user_id ↔ user.user_id

Many-to-One: user_refer_setting.updated_by ↔ admin.admin_id

15.Table: User_Refer_Transaction

- Columns:

  int id PK

  int user_id FK

  decimal credit_receive

  int refer_level

  int user_refer FK

  int payment_id FK

  datetime created_at

- Relationship:

Many-to-One: user_refer_transaction.user_refer ↔ user.user_id Many-to-One: user_refer_transaction.payment_id ↔payment.payment_id

16.Table: Trainer_Review

- Columns:

  int review_id PK

  int user_id FK

  int trainer_id FK

  int rating

  string comment

  datetime created_at

- Relationship:

Many-to-One: trainer_review.user_id ↔ user.user_id

Many-to-One: trainer_review.trainer_id ↔ trainer.trainer_id

17.Table: Coupon

- Columns:

  int coupon_id PK

  string code UK

  string name

  string description

  decimal discount

  datetime start_date

  datetime end_date

  int rank_id FK

  int created_by FK

  boolean is_active

- Relationship:

Many-to-One: coupon.rank_id ↔ rank.rank_id

Many-to-One: coupon.created_by ↔ admin.admin_id

SQL Statements

SQL Statement 1

Description: ดูประวัติการเข้าใช้งานยิมของสมาชิกแต่ละคนในเดือนมกราคม 2024

SQL Statement:

```sql
SELECT
    u.name,
    b.name as branch_name,
    COUNT(*) as visit_count,
    SUM(TIMESTAMPDIFF(MINUTE, g.check_in_time, g.check_out_time)) as total_minutes
FROM User u
JOIN Gym_Access_Log g ON u.user_id = g.user_id
JOIN Branch b ON g.branch_id = b.branch_id
WHERE MONTH(g.check_in_time) = 1 AND YEAR(g.check_in_time) = 2024
GROUP BY u.user_id, b.branch_id
ORDER BY total_minutes DESC;
```

Query Result:

	name	branch_name	visit_count	total_minutes
▶	Peter Wilson	Central Branch	1	120
	John Smith	Central Branch	1	120
	Mary Johnson	North Branch	1	120

SQL Statement 2

Description: คำนวณรายได้จาก Package แยกตามประเภทการชำระเงิน

SQL Statement:

```sql
SELECT
    p.payment_method,
    COUNT(*) as transaction_count,
    SUM(CASE
        WHEN c.code IS NOT NULL THEN p.amount * (1 - c.discount/100) ELSE p.amount
    END) as total_revenue,
    AVG(p.amount) as avg_amount,
    COUNT(c.code) as coupon_used_count
FROM Payment p
JOIN User_Package up ON p.payment_id = up.payment_id
JOIN Package pa ON up.package_id = pa.package_id
LEFT JOIN Coupon c ON p.coupon_code = c.code
WHERE YEAR(p.payment_date) = 2024
GROUP BY p.payment_method
ORDER BY total_revenue DESC;
```

Query Result:

	payment_method	transaction_count	total_revenue	avg_amount	coupon_used_count
▶	Credit	2	5350.00000000	2750.000000	1
	Credit Card	1	1500.00000000	1500.000000	0

SQL Statement 3

Description: แสดง Top 3 ผู้ใช้ที่มีคะแนนสะสมสูงสุด และชื่อ Rank

SQL Statement:

```sql
WITH RankedUsers AS (
    SELECT
        U.name AS user_name,
        U.rank_point,
        R.name AS rank_name,
        RANK() OVER (ORDER BY U.rank_point DESC) AS rank_position
    FROM `User` U
    JOIN `Rank` R ON U.rank_id = R.rank_id
)
SELECT *
FROM RankedUsers
WHERE rank_position <= 3;
```

Query Result:

	user_name	rank_point	rank_name	rank_position
▶	Peter Wilson	1500	Silver	1
	John Smith	500	Silver	2
	Mary Johnson	500	Silver	2

SQL Statement 4

Description: คำนวณค่าเฉลี่ยระยะเวลาการใช้บริการ (Check-in ถึง Check-out)

SQL Statement:

```sql
SELECT
    U.name AS user_name,
    AVG(TIMESTAMPDIFF(MINUTE, GAL.check_in_time, GAL.check_out_time)) AS avg_duration_minutes
FROM `Gym_Access_Log` GAL
JOIN `User` U ON GAL.user_id = U.user_id
GROUP BY GAL.user_id, U.name
ORDER BY avg_duration_minutes DESC;
```

Query Result:

	user_name	avg_duration_minutes
▶	John Smith	120.0000
	Mary Johnson	120.0000
	Peter Wilson	120.0000

SQL Statement 5

Description: หาผู้ใช้ที่เป็น Referral Master (ได้รับค่าคอมมิชชั่นสูงสุด)

SQL Statement:

```sql
SELECT
    U.name AS user_name,
    SUM(URT.credit_receive) AS total_earnings
FROM `User_Refer_Transaction` URT
JOIN `User` U ON URT.user_id = U.user_id
GROUP BY U.user_id, U.name
ORDER BY total_earnings DESC
LIMIT 1;
```

Query Result:

	user_name	total_earnings
▶	John Smith	275.00

Data Dictionary

Table: admin

Description: ข้อมูลผู้ดูแลระบบ

No.	Column Name	Description	Constraint	Referenced Table
1	admin_id	รหัสผู้ดูแลระบบ	NOT NULL PRIMARY KEY	
2	username	ชื่อผู้ใช้งานสำหรับเข้าระบบ	NOT NULL	
3	password	รหัสผ่าน	NOT NULL	
4	name	ชื่อ-นามสกุล	NOT NULL	
5	email	อีเมล	NOT NULL	
6	role	บทบาทในระบบ	NOT NULL	
7	last_login	วันเวลาที่เข้าสู่ระบบล่าสุด		
8	is_active	สถานะการใช้งาน	NOT NULL DEFAULT 1	

Table: branch

Description: ข้อมูลสาขา

No.	Column Name	Description	Constraint	Referenced Table
1	branch_id	รหัสสาขา	NOT NULL PRIMARY KEY	
2	name	ชื่อสาขา	NOT NULL	
3	phone	เบอร์โทรศัพท์	NOT NULL	
4	email	อีเมล	NOT NULL	
5	address	ที่อยู่	NOT NULL	
6	opening_time	เวลาเปิดทำการ	NOT NULL	
7	closing_time	เวลาปิดทำการ	NOT NULL	
8	status	สถานะการให้บริการ	NOT NULL	

Table: coupon

Description: ข้อมูลคูปองส่วนลด

No.	Column Name	Description	Constraint	Referenced Table
1	coupon_id	รหัสคูปอง	NOT NULL PRIMARY KEY	
2	code	รหัสส่วนลด	NOT NULL	
3	name	ชื่อโปรโมชั่น	NOT NULL	
4	description	รายละเอียดคูปอง		
5	discount	จำนวนส่วนลด	NOT NULL	
6	start_date	วันที่เริ่มใช้งานได้	NOT NULL	
7	end_date	วันที่หมดอายุ	NOT NULL	
8	rank_id	รหัสระดับสมาชิกที่ใช้ได้	FOREIGN KEY	rank
9	created_by	รหัสผู้สร้างคูปอง	FOREIGN KEY	admin
10	is_active	สถานะการใช้งาน	NOT NULL DEFAULT 1	
11	status	สถานะของคูปอง	NOT NULL	

Table: credit_history

Description: ประวัติการใช้เครดิต

No.	Column Name	Description	Constraint	Referenced Table
1	history_id	รหัสประวัติ	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	amount	จำนวนเครดิต	NOT NULL	
4	type	ประเภทรายการ	NOT NULL	
5	balance_before	ยอดก่อนทำรายการ	NOT NULL	
6	balance_after	ยอดหลังทำรายการ	NOT NULL	
7	created_at	วันเวลาที่ทำรายการ	NOT NULL	

Table: gym_access_log

Description: ประวัติการเข้าใช้งานฟิตเนส

No.	Column Name	Description	Constraint	Referenced Table
1	log_id	รหัสประวัติการเข้าใช้งาน	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	branch_id	รหัสสาขา	FOREIGN KEY NOT NULL	branch
4	user_package_id	รหัสแพ็คเกจที่ใช้งาน	FOREIGN KEY NOT NULL	user_package
5	check_in_time	เวลาเข้าใช้งาน	NOT NULL	
6	check_out_time	เวลาออกจากการใช้งาน		
7	status	สถานะการใช้งาน	NOT NULL	

Table: package

Description: ข้อมูลแพ็คเกจ

No.	Column Name	Description	Constraint	Referenced Table
1	package_id	รหัสแพ็คเกจ	NOT NULL PRIMARY KEY	
2	name	ชื่อแพ็คเกจ	NOT NULL	
3	description	รายละเอียดแพ็คเกจ		
4	price	ราคา	NOT NULL	
5	duration_days	ระยะเวลา (วัน)	NOT NULL	
6	earn_rank_point	คะแนนสะสมที่ได้รับ	NOT NULL	
7	is_active	สถานะการใช้งาน	NOT NULL DEFAULT 1	
8	created_by	รหัสผู้สร้างแพ็คเกจ	FOREIGN KEY	admin
9	created_at	วันที่สร้างข้อมูล	NOT NULL	
10	updated_at	วันที่แก้ไขข้อมูลล่าสุด	NOT NULL	
11	status	สถานะของแพ็คเกจ	NOT NULL	

Table: payment

Description: ข้อมูลการชำระเงิน

No.	Column Name	Description	Constraint	Referenced Table
1	payment_id	รหัสการชำระเงิน	NOT NULL PRIMARY KEY	
2	payment_method	วิธีการชำระเงิน	NOT NULL	
3	payment_date	วันเวลาที่ชำระเงิน	NOT NULL	
4	payment_status	สถานะการชำระเงิน	NOT NULL	
5	payment_note	บันทึกเพิ่มเติม		
6	amount	จำนวนเงิน	NOT NULL	
7	discount	ส่วนลด	NOT NULL DEFAULT 0	
8	total_amount	ยอดรวมสุทธิ	NOT NULL	
9	coupon_code	รหัสคูปองที่ใช้		
10	created_at	วันที่สร้างข้อมูล	NOT NULL	

Table: rank

Description: ข้อมูลระดับสมาชิก

No.	Column Name	Description	Constraint	Referenced Table
1	rank_id	รหัสระดับสมาชิก	NOT NULL PRIMARY KEY	
2	name	ชื่อระดับสมาชิก	NOT NULL	
3	min_point	คะแนนขั้นต่ำ	NOT NULL	
4	description	รายละเอียดระดับสมาชิก		

Table: topup_transaction

Description: ข้อมูลการเติมเงิน

No.	Column Name	Description	Constraint	Referenced Table
1	trans_id	รหัสธุรกรรม	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	status	สถานะการเติมเงิน	NOT NULL	
4	amount	จำนวนเงิน	NOT NULL	
5	payment_reference	รหัสอ้างอิงการชำระเงิน		
6	created_at	วันเวลาที่ทำรายการ	NOT NULL	

Table: trainer

Description: ข้อมูลเทรนเนอร์

No.	Column Name	Description	Constraint	Referenced Table
1	trainer_id	รหัสเทรนเนอร์	NOT NULL PRIMARY KEY	
2	name	ชื่อ-นามสกุล	NOT NULL	
3	description	รายละเอียดเทรนเนอร์		
4	specialization	ความเชี่ยวชาญ	NOT NULL	
5	base_salary	เงินเดือนพื้นฐาน	NOT NULL	
6	price_per_hour	ค่าบริการต่อชั่วโมง	NOT NULL	
7	status	สถานะการให้บริการ	NOT NULL	
8	branch_id	รหัสสาขา	FOREIGN KEY NOT NULL	branch
9	address	ที่อยู่		
10	joined_date	วันที่เริ่มงาน	NOT NULL	

Table: trainer_booking

Description: ข้อมูลการจองเทรนเนอร์

No.	Column Name	Description	Constraint	Referenced Table
1	booking_id	รหัสการจอง	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	trainer_id	รหัสเทรนเนอร์	FOREIGN KEY NOT NULL	trainer
4	branch_id	รหัสสาขา	FOREIGN KEY NOT NULL	branch
5	booking_date	วันที่จอง	NOT NULL	
6	start_time	เวลาเริ่มต้น	NOT NULL	
7	end_time	เวลาสิ้นสุด	NOT NULL	
8	hours	จำนวนชั่วโมง	NOT NULL	
9	price_per_hour	ราคาต่อชั่วโมง	NOT NULL	
10	total_amount	ยอดรวม	NOT NULL	
11	status	สถานะการจอง	NOT NULL	
12	cancellation_reason	เหตุผลการยกเลิก		
13	payment_id	รหัสการชำระเงิน	FOREIGN KEY	payment
14	created_at	วันที่สร้างข้อมูล	NOT NULL	

Table: trainer_review

Description: ข้อมูลการรีวิวเทรนเนอร์

No.	Column Name	Description	Constraint	Referenced Table
1	review_id	รหัสรีวิว	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	trainer_id	รหัสเทรนเนอร์	FOREIGN KEY NOT NULL	trainer
4	rating	คะแนน	NOT NULL	
5	comment	ความคิดเห็น		
6	created_at	วันที่รีวิว	NOT NULL	

Table: trainer_schedule

Description: ตารางงานเทรนเนอร์

No.	Column Name	Description	Constraint	Referenced Table
1	schedule_id	รหัสตารางงาน	NOT NULL PRIMARY KEY	
2	trainer_id	รหัสเทรนเนอร์	FOREIGN KEY NOT NULL	trainer
3	work_date	วันที่ทำงาน	NOT NULL	
4	start_time	เวลาเริ่มงาน	NOT NULL	
5	end_time	เวลาเลิกงาน	NOT NULL	
6	status	สถานะการทำงาน	NOT NULL	

Table: user_package

Description: ข้อมูลแพ็คเกจของผู้ใช้งาน

No.	Column Name	Description	Constraint	Referenced Table
1	id	รหัสแพ็คเกจผู้ใช้งาน	NOT NULL PRIMARY KEY	
2	user_id	รหัสผู้ใช้งาน	FOREIGN KEY NOT NULL	user
3	package_id	รหัสแพ็คเกจ	FOREIGN KEY NOT NULL	package
4	payment_id	รหัสการชำระเงิน	FOREIGN KEY NOT NULL	payment
5	start_date	วันที่เริ่มใช้งาน	NOT NULL	
6	exp_date	วันที่หมดอายุ	NOT NULL	
7	status	สถานะการใช้งาน	NOT NULL	
8	earn_rank_point	คะแนนสะสมที่ได้รับ	NOT NULL	

Table: user_refer_setting

Description: การตั้งค่าการแนะนำสมาชิก

No.	Column Name	Description	Constraint	Referenced Table
1	user_id	รหัสผู้ใช้งาน	PRIMARY KEY, FOREIGN KEY	user
2	level_1	เปอร์เซ็นต์ผลตอบแทนระดับที่ 1	NOT NULL	
3	level_2	เปอร์เซ็นต์ผลตอบแทนระดับที่ 2	NOT NULL	
4	level_3	เปอร์เซ็นต์ผลตอบแทนระดับที่ 3	NOT NULL	
5	level_4	เปอร์เซ็นต์ผลตอบแทนระดับที่ 4	NOT NULL	
6	created_at	วันที่สร้างข้อมูล	NOT NULL	
7	updated_by	รหัสผู้แก้ไขข้อมูล	FOREIGN KEY	admin

Table: user_refer_transaction

Description: ธุรกรรมการได้รับผลตอบแทนจากการแนะนำ

No.	Column Name	Description	Constraint	Referenced Table
1	id	รหัสธุรกรรม	NOT NULL PRIMARY KEY	
2	credit_receive	จำนวนเครดิตที่ได้รับ	NOT NULL	
3	refer_level	ระดับการแนะนำ	NOT NULL	
4	user_refer	รหัสผู้แนะนำ	FOREIGN KEY NOT NULL	user
5	payment_id	รหัสการชำระเงิน	FOREIGN KEY NOT NULL	payment
6	created_at	วันที่สร้างข้อมูล	NOT NULL	

DDL Script for Creating the Database

```sql
-- Drop database if exists and create new one
DROP DATABASE IF EXISTS phalo_fitness;
CREATE DATABASE phalo_fitness;
USE phalo_fitness;

-- สร้างตาราง Rank
CREATE TABLE `rank` (
    rank_id INT PRIMARY KEY,
    name VARCHAR(255),
    min_point INT,
    description TEXT
);

-- สร้างตาราง Payment ขึ้นมาก่อน เนื่องจากมีตารางอื่นอ้างอิง
CREATE TABLE `payment` (
    payment_id INT PRIMARY KEY,
    payment_method VARCHAR(50),
    payment_date DATETIME,
    payment_status VARCHAR(50),
    payment_note TEXT,
    amount DECIMAL(10,2),
    discount DECIMAL(10,2),
    total_amount DECIMAL(10,2),
    coupon_code VARCHAR(255),
    created_at DATETIME
);

-- สร้างตาราง User
CREATE TABLE `user` (
    user_id INT PRIMARY KEY,
    username VARCHAR(255) UNIQUE,
    password VARCHAR(255),
    credit DECIMAL(10,2),
    name VARCHAR(255),
    email VARCHAR(255) UNIQUE,
    phone VARCHAR(50),
    address VARCHAR(255),
    rank_point INT,
    user_refer INT,
    rank_id INT,
```

```sql
    created_at DATETIME,
    updated_at DATETIME,
    is_active BOOLEAN,
    FOREIGN KEY (user_refer) REFERENCES `User`(user_id),
    FOREIGN KEY (rank_id) REFERENCES `Rank`(rank_id)
);

-- สร้างตาราง Admin
CREATE TABLE `admin` (
    admin_id INT PRIMARY KEY,
    username VARCHAR(255) UNIQUE,
    password VARCHAR(255),
    name VARCHAR(255),
    email VARCHAR(255) UNIQUE,
    role VARCHAR(50),
    last_login DATETIME,
    is_active BOOLEAN
);

-- สร้างตาราง Package
CREATE TABLE `package` (
    package_id INT PRIMARY KEY,
    name VARCHAR(255),
    description TEXT,
    price DECIMAL(10,2),
    duration_days INT,
    earn_rank_point INT,
    is_active BOOLEAN,
    created_by INT,
    created_at DATETIME,
    updated_at DATETIME,
    status VARCHAR(30),
    FOREIGN KEY (created_by) REFERENCES `Admin`(admin_id)
);

-- สร้างตาราง Coupon
CREATE TABLE `coupon` (
    coupon_id INT PRIMARY KEY,
    code VARCHAR(255) UNIQUE,
    name VARCHAR(255),
    description TEXT,
    discount DECIMAL(10,2),
    start_date DATETIME,
```

```sql
    end_date DATETIME,
    rank_id INT,
    created_by INT,
    is_active BOOLEAN,
   status VARCHAR(50),
    FOREIGN KEY (rank_id) REFERENCES `Rank`(rank_id),
    FOREIGN KEY (created_by) REFERENCES `Admin`(admin_id)
);

-- อัพเดท Foreign Key สำหรับ Payment หลังจากสร้างตาราง Coupon
ALTER TABLE `payment`
ADD FOREIGN KEY (coupon_code) REFERENCES `Coupon`(code);

-- สร้างตาราง User_Package
CREATE TABLE `user_package` (
    id INT PRIMARY KEY,
    user_id INT,
    package_id INT,
    payment_id INT,
    start_date DATETIME,
    exp_date DATETIME,
    status VARCHAR(50),
    earn_rank_point INT,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (package_id) REFERENCES `Package`(package_id),
    FOREIGN KEY (payment_id) REFERENCES `Payment`(payment_id)
);

-- สร้างตาราง Branch
CREATE TABLE `branch` (
    branch_id INT PRIMARY KEY,
    name VARCHAR(255),
    phone VARCHAR(50),
    email VARCHAR(255),
    address VARCHAR(255),
    status INT,
    opening_time TIME,
    closing_time TIME
);

-- สร้างตาราง Trainer
CREATE TABLE `trainer` (
    trainer_id INT PRIMARY KEY,
```

```
    name VARCHAR(255),
    description TEXT,
    specialization VARCHAR(255),
    base_salary DECIMAL(10,2),
    price_per_hour DECIMAL(10,2),
    status VARCHAR(50),
    branch_id INT,
    address VARCHAR(255),
    joined_date DATETIME,
    FOREIGN KEY (branch_id) REFERENCES `Branch`(branch_id)
);



-- สร้างตาราง Trainer_Schedule
CREATE TABLE `trainer_schedule` (
    schedule_id INT PRIMARY KEY,
    trainer_id INT,
    work_date DATE,
    start_time TIME,
    end_time TIME,
    status VARCHAR(50),
    FOREIGN KEY (trainer_id) REFERENCES `Trainer`(trainer_id)
);

-- สร้างตาราง Trainer_Booking
CREATE TABLE `trainer_booking` (
    booking_id INT PRIMARY KEY,
    user_id INT,
    trainer_id INT,
    branch_id INT,
    booking_date DATETIME,
    start_time TIME,
    end_time TIME,
    hours INT,
    price_per_hour DECIMAL(10,2),
    total_amount DECIMAL(10,2),
    status VARCHAR(50),
    cancellation_reason TEXT,
    payment_id INT,
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (trainer_id) REFERENCES `Trainer`(trainer_id),
```

```
    FOREIGN KEY (branch_id) REFERENCES `Branch`(branch_id),
    FOREIGN KEY (payment_id) REFERENCES `Payment`(payment_id)
);

-- สร้างตาราง Gym_Access_Log
CREATE TABLE `gym_access_log` (
    log_id INT PRIMARY KEY,
    user_id INT,
    branch_id INT,
    user_package_id INT,
    check_in_time DATETIME,
    check_out_time DATETIME,
    status VARCHAR(50),
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (branch_id) REFERENCES `Branch`(branch_id),
    FOREIGN KEY (user_package_id) REFERENCES `User_Package`(id)
);


-- สร้างตาราง Credit_History
CREATE TABLE `credit_history` (
    history_id INT PRIMARY KEY,
    user_id INT,
    amount DECIMAL(10,2),
    type VARCHAR(50),
    balance_before DECIMAL(10,2),
    balance_after DECIMAL(10,2),
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id)
);

-- สร้างตาราง Topup_Transaction
CREATE TABLE `topup_transaction` (
    trans_id INT PRIMARY KEY,
    user_id INT,
    status VARCHAR(50),
    amount DECIMAL(10,2),
    payment_reference VARCHAR(255),
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id)
);

-- สร้างตาราง User_Refer_Setting
```

```sql
CREATE TABLE `user_refer_setting` (
    user_id INT PRIMARY KEY,
    level_1 DECIMAL(5,2),
    level_2 DECIMAL(5,2),
    level_3 DECIMAL(5,2),
    level_4 DECIMAL(5,2),
    created_at DATETIME,
    updated_by INT,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (updated_by) REFERENCES `Admin`(admin_id)
);

-- สร้างตาราง User_Refer_Transaction
CREATE TABLE `user_refer_transaction` (
    id INT PRIMARY KEY,
    user_id INT,
    credit_receive DECIMAL(10,2),
    refer_level INT,
    user_refer INT,
    payment_id INT,
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (user_refer) REFERENCES `User`(user_id),
    FOREIGN KEY (payment_id) REFERENCES `Payment`(payment_id)
);

-- สร้างตาราง Trainer_Review
CREATE TABLE `trainer_review` (
    review_id INT PRIMARY KEY,
    user_id INT,
    trainer_id INT,
    rating INT,
    comment TEXT,
    created_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES `User`(user_id),
    FOREIGN KEY (trainer_id) REFERENCES `Trainer`(trainer_id)
);

-- สร้าง ENUM สำหรับแต่ละตาราง

-- Admin Status
ALTER TABLE `Admin`
MODIFY COLUMN role ENUM('admin', 'manager', 'staff') NOT NULL DEFAULT 'staff';
```

```sql
-- Package Status
ALTER TABLE `Package`
MODIFY COLUMN status ENUM('active', 'inactive', 'discontinued') NOT NULL DEFAULT 'active';

-- User_Package Status
ALTER TABLE `User_Package`
MODIFY COLUMN status ENUM(
    'active',        -- แพ็คเกจกำลังใช้งาน
    'expired',       -- แพ็คเกจหมดอายุ
    'cancelled',     -- ยกเลิกก่อนหมดอายุ
    'suspended',     -- ระงับชั่วคราว
    'pending'        -- รอการชำระเงิน
) NOT NULL DEFAULT 'pending';

-- Branch Status
ALTER TABLE `Branch`
MODIFY COLUMN status ENUM('active', 'inactive', 'renovating', 'closed') NOT NULL DEFAULT 'active';

-- Trainer Status
ALTER TABLE `Trainer`
MODIFY COLUMN status ENUM(
    'available',     -- พร้อมให้บริการ
    'booked',        -- จองแล้ว
    'unavailable',   -- ไม่ว่าง/ลา
    'training',      -- กำลังสอน
    'terminated'     -- พ้นสภาพ
) NOT NULL DEFAULT 'available';


-- Trainer_Schedule Status
ALTER TABLE `Trainer_Schedule`
MODIFY COLUMN status ENUM(
    'available',     -- ว่างให้จอง
    'booked',        -- มีการจองแล้ว
    'completed',     -- เสร็จสิ้นการสอน
    'cancelled',     -- ยกเลิก
    'no_show'        -- ลูกค้าไม่มา
) NOT NULL DEFAULT 'available';

-- Payment Status
ALTER TABLE `Payment`
MODIFY COLUMN payment_status ENUM(
```

```
    'pending',        -- รอการชำระเงิน
    'processing',     -- กำลังดำเนินการ
    'completed',      -- ชำระเงินสำเร็จ
    'failed',         -- การชำระเงินล้มเหลว
    'refunded',       -- คืนเงินแล้ว
    'cancelled'       -- ยกเลิก
) NOT NULL DEFAULT 'pending',
MODIFY COLUMN payment_method ENUM(
    'credit',         -- เครดิตในระบบ
    'credit_card',    -- บัตรเครดิต
    'debit_card',     -- บัตรเดบิต
    'bank_transfer', -- โอนเงิน
    'qr_code',        -- QR Payment
    'cash'            -- เงินสด
) NOT NULL;

-- Trainer_Booking Status
ALTER TABLE `Trainer_Booking`
MODIFY COLUMN status ENUM(
    'pending',        -- รอการยืนยัน
    'confirmed',      -- ยืนยันการจองแล้ว
    'in_progress',    -- กำลังเทรน
    'completed',      -- เสร็จสิ้น
    'cancelled',      -- ยกเลิก
    'no_show',        -- ลูกค้าไม่มา
    'rescheduled'     -- เลื่อนการจอง
) NOT NULL DEFAULT 'pending';

-- Gym_Access_Log Status
ALTER TABLE `Gym_Access_Log`
MODIFY COLUMN status ENUM(
    'checked_in',     -- เช็คอินแล้ว
    'checked_out',    -- เช็คเอาท์แล้ว
    'in_progress',    -- อยู่ในยิม
    'cancelled',      -- ยกเลิก
    'expired',        -- แพ็คเกจหมดอายุ
    'invalid'         -- ไม่ถูกต้อง
) NOT NULL DEFAULT 'checked_in';

-- Credit_History Type
ALTER TABLE `Credit_History`
MODIFY COLUMN type ENUM(
    'topup',              -- เติมเงิน
```

```
    'purchase_package',  -- ซื้อแพ็คเกจ
    'purchase_trainer',  -- จ่ายค่าเทรนเนอร์
    'refund_package',    -- คืนเงินค่าแพ็คเกจ
    'refund_trainer',    -- คืนเงินค่าเทรนเนอร์
    'referral_bonus',    -- โบนัสแนะนำ
    'adjustment_add',    -- ปรับยอดเพิ่ม
    'adjustment_deduct', -- ปรับยอดลด
    'expired'            -- เครดิตหมดอายุ
) NOT NULL;

-- Topup_Transaction Status
ALTER TABLE `Topup_Transaction`
MODIFY COLUMN status ENUM(
    'pending',       -- รอการชำระเงิน
    'processing',    -- กำลังดำเนินการ
    'completed',     -- สำเร็จ
    'failed',        -- ล้มเหลว
    'cancelled',     -- ยกเลิก
    'refunded'       -- คืนเงิน
) NOT NULL DEFAULT 'pending';

-- Coupon Status
ALTER TABLE `Coupon`
MODIFY COLUMN status ENUM(
    'active',        -- ใช้งานได้
    'inactive',      -- ปิดการใช้งาน
    'expired',       -- หมดอายุ
    'depleted'       -- ใช้ครบจำนวนแล้ว
) NOT NULL DEFAULT 'active';
```

Export Data in the format of INSERT statements

```
INSERT INTO `Rank` (rank_id, name, min_point, description) VALUES
(1, 'Bronze', 0, 'ระดับเริ่มต้นสำหรับสมาชิกใหม่'),
(2, 'Silver', 1000, 'สำหรับสมาชิกที่มีคะแนนสะสมมากกว่า 1000'),
(3, 'Gold', 5000, 'สำหรับสมาชิกที่มีคะแนนสะสมมากกว่า 5000'),
(4, 'Platinum', 10000, 'สำหรับสมาชิกที่มีคะแนนสะสมมากกว่า 10000');
```

1. Insert into Rank
2. Insert into Admin

```
INSERT INTO `Admin` (admin_id, username, password, name, email, role, last_login, is_active) VALUES
(1, 'admin1', 'hashed_password1', 'Admin One', 'admin1@example.com', 'Admin', '2024-01-01 08:00:00', TRUE),
(2, 'admin2', 'hashed_password2', 'Admin Two', 'admin2@example.com', 'Manager', '2024-01-02 09:00:00', TRUE);
```

3. Insert into Users (เริ่มด้วยเครดิต 0)

```
INSERT INTO `User` (user_id, username, password, credit, name, email, phone, address, rank_point, user_refer, rank_id, created_at,
updated_at, is_active) VALUES
(1, 'john123', 'hashed_password1', 0.00, 'John Smith', 'john@example.com', '0812345678', '123 Main St', 0, NULL, 1, '2024-01-01
10:00:00', '2024-01-01 10:00:00', TRUE),
(2, 'mary456', 'hashed_password2', 0.00, 'Mary Johnson', 'mary@example.com', '0823456789', '456 Second St', 0, 1, 1, '2024-01-02
11:00:00', '2024-01-02 11:00:00', TRUE),
(3, 'peter789', 'hashed_password3', 0.00, 'Peter Wilson', 'peter@example.com', '0834567890', '789 Third St', 0, 1, 1, '2024-01-03
12:00:00', '2024-01-03 12:00:00', TRUE);
```

4. Insert Topup Transactions (การเติมเงินเข้าระบบ)

```
INSERT INTO `Topup_Transaction` (trans_id, user_id, status, amount, payment_reference, created_at) VALUES
(1, 1, 'Completed', 5000.00, 'TOP24010101', '2024-01-01 10:30:00'),
(2, 2, 'Completed', 3000.00, 'TOP24010201', '2024-01-02 11:30:00'),
(3, 3, 'Completed', 2000.00, 'TOP24010301', '2024-01-03 12:30:00');
```

5. Insert Package

```
INSERT INTO `Package` (package_id, name, description, price, duration_days, earn_rank_point, is_active, created_by, created_at, updated_at) VALUES
(1, 'Basic Monthly', 'แพ็คเกจรายเดือน', 1500.00, 30, 500, TRUE, 1, '2024-01-01', '2024-01-01'),
(2, 'Premium Quarterly', 'แพ็คเกจ 3 เดือน', 4000.00, 90, 1500, TRUE, 1, '2024-01-01', '2024-01-01');
```

6. Insert Coupon

```
INSERT INTO `Payment` (payment_id, payment_method, payment_date, payment_status, payment_note, amount, discount, total_amount, coupon_code, created_at) VALUES
-- John ซื้อ Basic Package ด้วยเครดิตในระบบ ใช้คูปอง
(1, 'Credit', '2024-01-01 11:00:00', 'completed', 'Basic Monthly Package', 1500.00, 150.00, 1350.00, 'NEW2024', '2024-01-01 11:00:00'),
-- Mary ซื้อ Basic Package ด้วยบัตรเครดิต
(2, 'credit_card', '2024-01-02 12:00:00', 'completed', 'Basic Monthly Package', 1500.00, 0.00, 1500.00, NULL, '2024-01-02 12:00:00'),
-- Peter ซื้อ Premium Package ด้วยเครดิตในระบบ
```

7. Insert Payments (สามารถจ่ายด้วยเครดิตในระบบหรือช่องทางอื่นๆ)

```
INSERT INTO `Payment` (payment_id, payment_method, payment_date, payment_status, payment_note, amount, discount, total_amount, coupon_code, created_at) VALUES
-- John ซื้อ Basic Package ด้วยเครดิตในระบบ ใช้คูปอง
(1, 'Credit', '2024-01-01 11:00:00', 'completed', 'Basic Monthly Package', 1500.00, 150.00, 1350.00, 'NEW2024', '2024-01-01 11:00:00'),
-- Mary ซื้อ Basic Package ด้วยบัตรเครดิต
(2, 'credit_card', '2024-01-02 12:00:00', 'completed', 'Basic Monthly Package', 1500.00, 0.00, 1500.00, NULL, '2024-01-02 12:00:00'),
-- Peter ซื้อ Premium Package ด้วยเครดิตในระบบ
(3, 'Credit', '2024-01-03 13:00:00', 'completed', 'Premium Quarterly Package', 4000.00, 0.00, 4000.00, NULL, '2024-01-03 13:00:00');
```

8. Record Credit History for credit payments

```
INSERT INTO `Credit_History` (history_id, user_id, amount, type, balance_before, balance_after, created_at) VALUES
-- John's transactions
(1, 1, 5000.00, 'topup', 0.00, 5000.00, '2024-01-01 10:30:00'),
(2, 1, -1350.00, 'purchase_package', 5000.00, 3650.00, '2024-01-01 11:00:00'),
-- Mary's transactions
(3, 2, 3000.00, 'topup', 0.00, 3000.00, '2024-01-02 11:30:00'),
-- Peter's transactions
(4, 3, 2000.00, 'topup', 0.00, 2000.00, '2024-01-03 12:30:00'),
(5, 3, -4000.00, 'purchase_package', 2000.00, -2000.00, '2024-01-03 13:00:00');
```

9. Insert User Packages

```
INSERT INTO `User_Package` (id, user_id, package_id, payment_id, start_date, exp_date, status, earn_rank_point) VALUES
(1, 1, 1, 1, '2024-01-01', '2024-01-31', 'active', 500),
(2, 2, 1, 2, '2024-01-02', '2024-02-01', 'active', 500),
(3, 3, 2, 3, '2024-01-03', '2024-04-02', 'active', 1500);
```

10. Setup Referral Settings

```
INSERT INTO `User_Refer_Setting` (user_id, level_1, level_2, level_3, level_4, created_at, updated_by) VALUES
(1, 5.00, 3.00, 2.00, 1.00, '2024-01-01', 1);
```

11. Record Referral Transactions

```
INSERT INTO `User_Refer_Transaction` (id, user_id, credit_receive, refer_level, user_refer, payment_id, created_at) VALUES
-- John (id:1) gets 5% from Mary's (id:2) purchase
(1, 1, 75.00, 1, 2, 2, '2024-01-02 12:01:00'),
-- John (id:1) gets 5% from Peter's (id:3) purchase
(2, 1, 200.00, 1, 3, 3, '2024-01-03 13:01:00');
```

12. Record Credit History for referral earnings

```
INSERT INTO `Credit_History` (history_id, user_id, amount, type, balance_before, balance_after, created_at) VALUES
(6, 1, 75.00, 'referral_bonus', 3650.00, 3725.00, '2024-01-02 12:01:00'),
(7, 1, 200.00, 'referral_bonus', 3725.00, 3925.00, '2024-01-03 13:01:00');
```

13. Insert Branch

```
INSERT INTO `Branch` (branch_id, name, phone, email, address, opening_time, closing_time, status) VALUES
(1, 'Central Branch', '021234567', 'central@example.com', '123 Main St', '06:00:00', '22:00:00', 'active'),
(2, 'North Branch', '022345678', 'north@example.com', '456 North St', '06:00:00', '22:00:00', 'active');
```

14. Insert Trainer

```
INSERT INTO `Trainer` (trainer_id, name, description, specialization, base_salary, price_per_hour, status, branch_id, address,
joined_date) VALUES
(1, 'Mike Johnson', 'Certified Personal Trainer', 'Weight Training', 30000.00, 800.00, 'Available', 1, '789 Trainer St', '2024-01-01'),
(2, 'Sarah Smith', 'Yoga Specialist', 'Yoga and Flexibility', 28000.00, 700.00, 'Available', 2, '101 Yoga St', '2024-01-01');
```

15. Insert Gym Access Logs

```
INSERT INTO `Gym_Access_Log` (log_id, user_id, branch_id, user_package_id, check_in_time, check_out_time, status) VALUES
(1, 1, 1, 1, '2024-01-02 09:00:00', '2024-01-02 11:00:00', 'checked_out'),
(2, 2, 2, 2, '2024-01-03 15:00:00', '2024-01-03 17:00:00', 'checked_out'),
(3, 3, 1, 3, '2024-01-04 10:00:00', '2024-01-04 12:00:00', 'checked_out');
```