Fitness Tracker

1. Teams

- นาย ทาคูยะ มาเอกาวา รหัสนักศึกษา 65123853
- นาย สุภัทรถณ ตรีทอง รหัสนักศึกษา 65132706
- นางสาว นภัสสร ใจแผ้ว รหัสนักศึกษา 65104192

2. Introduction (5 คะแนน, -0.25)

a. Problem Statment

ในปัจจุบันผู้ทคนสนใจการออกกำลังกายจำนวนมาก มีความต้องการที่จะติดตามและบันทึกข้อมูลการออกกำลังกาย เช่น ระยะทางที่ออกกำลังกายและอัตราการเต้นของหัวใจ (BPM) แต่การบันทึกข้อมูลเหล่านี้อาจเป็นเรื่องที่ยุ่งยากและซับซ้อนหากไม่มีเครื่องมือที่เหมาะสม นอกจากนี้ การแสดงผลข้อมูลในรูปแบบที่เป็นภาพรวมของการออกกำลังกาย เช่น การดูอัตราการเต้นของหัวใจโดยเฉลี่ย ก็เป็นสิ่งที่ทำได้ยากหากไม่มีระบบสนับสนุน

ดังนั้นจึงมีความจำเป็นในการพัฒนาระบบบันทึกการออกกำลังกายที่ให้ผู้ใช้สามารถจดบันทึกระยะทางการออกกำลังกายและอัตราการเต้นของหัวใจได้ด้วยตนเองทุกครั้งที่ออกกำลังกาย และมีการแสดงผลที่สามารถแสดงผลข้อมูลเหล่านี้ในรูปแบบที่เข้าใจง่าย เพื่อให้ผู้ใช้สามารถติดตามความก้าวหน้า ปรับปรุงแผนการออกกำลังกาย และเพิ่มประสิทธิภาพในการดูแลสุขภาพของตนเองได้อย่างต่อเนื่อง

b. Objectives

1. พัฒนาระบบที่ให้ผู้ใช้สามารถจดบันทึกระยะทางที่ออกกำลังกายและอัตราการเต้นของหัวใจ (BPM) ได้ด้วยตนเองทุกครั้ง โดยมุ่งเน้นให้การบันทึกข้อมูลทำได้อย่างรวดเร็วและไม่ยุ่งยาก

2. สร้างฟังก์ชันที่สามารถแสดงผลข้อมูลการออกกำลังกาย เช่น อัตราการเต้นของหัวใจเฉลี่ย ระยะทางที่ออกกำลังกาย ด้วยกราฟหรือสถิติที่ง่ายต่อการเข้าใจ

3. ออกแบบระบบที่ช่วยให้ผู้ใช้สามารถตรวจสอบความก้าวหน้าในการออกกำลังกายได้ เช่น การเปรียบเทียบข้อมูลจากการออกกำลังกายครั้งก่อน ๆ เพื่อปรับแผนการออกกำลังกายให้เหมาะสม

3. Design

a. System Architecture (5 คะแนน, -0.25)

1. **Presentation Layer (Frontend)**:

   - หน้าจอสำหรับกรอกข้อมูลระยะทางและอัตราการเต้นของหัวใจ
   - หน้าจอแสดงผลประวัติการออกกำลังกาย เช่น ระยะทาง, อัตราการเต้นของหัวใจ, วันเวลา ฯลฯ
   - การนำทางไปยังหน้าจอต่าง ๆ และระบบแจ้งเตือนเมื่อมีข้อมูลไม่ถูกต้อง

2. **Business Logic Layer (Application Layer)**:

   - การคำนวณสถิติการออกกำลังกาย เช่น เฉลี่ยอัตราการเต้นของหัวใจ, การเผาผลาญแคลอรี่ เป็นต้น

   - การตรวจสอบความถูกต้องของข้อมูลที่ผู้ใช้กรอกเข้ามา เช่น ระยะทางและอัตราการเต้นของหัวใจ

   - การจัดการผู้ใช้ เช่น การสมัครสมาชิก, การเข้าสู่ระบบ, การรีเซ็ตรหัสผ่าน

3. **Data Layer**:

   - การจัดเก็บข้อมูลผู้ใช้ ข้อมูลการออกกำลังกาย และสถิติที่เกี่ยวข้อง
   - การซิงค์ข้อมูลกับ Cloud Storage หรือฐานข้อมูลที่อยู่บนเซิร์ฟเวอร์
   - การแคชข้อมูลเพื่อให้เข้าถึงได้รวดเร็ว

4. **Networking Layer**:

   - การเชื่อมต่อและส่งข้อมูลระหว่างแอปพลิเคชันและเซิร์ฟเวอร์
   - การรับ-ส่งข้อมูลผ่าน API กับบริการภายนอก
   - การจัดการกับการตอบสนองและการประมวลผลผลลัพธ์ที่ได้รับจากเซิร์ฟเวอร์

5. **Third-Party Services Integration**:

   - การจัดการระบบการแจ้งเตือน

- การผสานรวมกับบริการคลาวด์สำหรับจัดเก็บข้อมูลหรือประมวลผลเพิ่มเติม
- การเชื่อมต่อกับโซเชียลมีเดียเพื่อแชร์ข้อมูลการออกกำลังกาย

6. **Development and Deployment Environment**:

- ระบบจัดการเวอร์ชันของโค้ดและการทำงานร่วมกันของทีมพัฒนา
- การตั้งค่าเซิร์ฟเวอร์สำหรับการปรับใช้แอปพลิเคชันไปยัง Production Environment
- การจัดการกระบวนการ CI/CD เพื่อทำการ Build, Test และ Deploy โดยอัตโนมัติ

b. Software Architecture (5 คะแนน, -0.25)

7. **UI Components:**

- Buttons: ออกแบบเป็นองค์ประกอบที่สามารถปรับแต่งได้ เช่น สี รูปร่าง ขนาด และการจัดวาง โดยใช้ UI Framework ที่รองรับการกำหนดค่า หรือธีม เพื่อให้สามารถนำไปใช้ซ้ำได้ในส่วนต่าง ๆ ของแอปพลิเคชัน
- TextFields: ใช้สำหรับรับข้อมูลจากผู้ใช้ โดยสามารถตั้งค่าคุณสมบัติ เช่น การตรวจสอบความถูกต้องของข้อมูล การแสดงผลข้อความอธิบาย และการจัดการกับเหตุการณ์ เพื่อให้สามารถปรับแต่งและนำไปใช้ซ้ำได้ง่าย
- Lists: ใช้สำหรับแสดงรายการข้อมูล เช่น ประวัติการออกกำลังกาย โดยใช้ส่วนประกอบที่สามารถแสดงรายการแบบ Dynamic List ซึ่งรองรับการออกแบบที่สามารถปรับแต่งได้ และสามารถนำกลับมาใช้ซ้ำได้
- Customizable and Reusable: ใช้แนวคิดของ Atomic Design หรือ Component-Based Architecture ในการออกแบบ UI เพื่อให้สามารถแบ่งส่วนประกอบต่าง ๆ ออกเป็นหน่วยย่อย ๆ ที่สามารถปรับแต่งได้และนำกลับมาใช้ซ้ำได้ในหลาย ๆ หน้าจอ

8. **Data Management:**

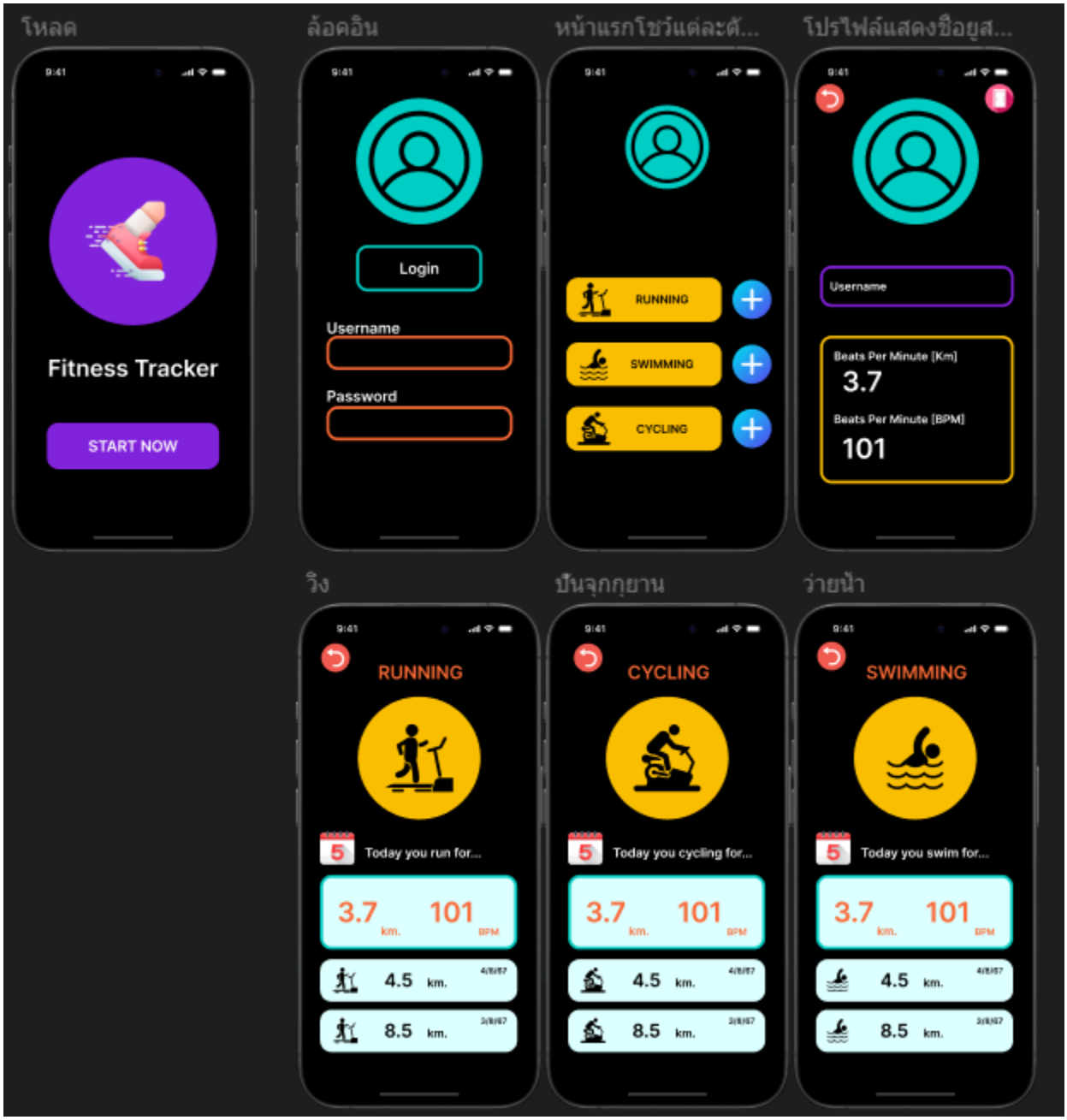
- Local Database: ใช้ Realm ในการจัดเก็บข้อมูลภายในเครื่อง เช่น ประวัติการออกกำลังกาย อัตราการเต้นของหัวใจ โดยอาจมีการใช้ Encryption เพื่อเพิ่มความปลอดภัยให้กับข้อมูลที่สำคัญ
- State Management: ใช้ State Management Libraries เพื่อจัดการสถานะของข้อมูลในแอปพลิเคชัน เพื่อให้การแสดงผลข้อมูลกับการประมวลผลเป็นไปอย่างราบรื่นและมีประสิทธิภาพ

- Repository Pattern: ใช้ Repository Pattern เพื่อแยกการจัดการแหล่งข้อมูลออกจาก Business Logic โดยมีการจัดการข้อมูลจากหลายแหล่ง เช่น Local Database, Remote API โดยให้ Repository ทำหน้าที่ประสานข้อมูลจากแหล่งต่าง ๆ และส่งข้อมูลที่สะอาดให้กับ Application Layer

9. **Networking:**

- API Calls: ใช้ HTTP Client Libraries ในการเรียกใช้งาน API โดยออกแบบให้รองรับการเรียกใช้งานแบบ Asynchronous เพื่อไม่ให้ UI ถูกบล็อกระหว่างการดึงข้อมูล
- Real-Time Data Sync: ใช้เทคโนโลยีเช่น WebSocket หรือ Firebase Realtime Database สำหรับการดึงข้อมูลแบบ Real-Time เพื่อให้สามารถแสดงข้อมูลอัปเดตล่าสุดให้กับผู้ใช้ได้ทันที เช่น การอัปเดตสถิติการออกกำลังกายทันทีเมื่อมีการบันทึกข้อมูลใหม่
- Error Handling & Retry Mechanism: จัดการกับข้อผิดพลาดในการเชื่อมต่อเครือข่าย เช่น Timeout, Server Unreachable โดยมีระบบ Retry ที่สามารถลองดึงข้อมูลใหม่อีกครั้งเมื่อเกิดข้อผิดพลาด นอกจากนี้ยังสามารถแสดงสถานะการเชื่อมต่อให้กับผู้ใช้ เช่น "กำลังเชื่อมต่อ" "ไม่มีอินเทอร์เน็ต"

c. UX/UI Design (5 คะแนน, -0.25)



หน้าจอโหลด

UX: ออกแบบมาให้ผู้ใช้เห็นภาพรวมของแอปพลิเคชันได้ทันที โดยมีไอคอนรองเท้าวิ่งที่สื่อถึงการออกกำลังกาย พร้อมปุ่ม START NOW เพื่อไปหน้าจอ Login

UI: ใช้สีสันที่สดใสและตัวอักษรที่ชัดเจน ทำให้ดูน่าสนใจ กระตุ้นให้เกิดความ active และปุ่ม "START NOW" มีขนาดใหญ่และเด่นชัด เพื่อกระตุ้นให้ผู้ใช้เริ่มต้นใช้งาน

หน้าจอเข้าสู่ระบบ (Login)

UX: กระบวนการเข้าสู่ระบบที่ง่าย โดยมีช่องสำหรับกรอก Username และ Password เพียงสองช่อง พร้อมให้สามารถกดปุ่ม Login เพื่อเข้าใช้งานหน้าต่อไป

UI: การจัดวางองค์ประกอบต่างๆ บนหน้าจอมีความสมดุล และปุ่ม "Login" มีขนาดใหญ่และเด่นชัด

หน้าจอหลัก (Home)

UX: แสดงรายการกิจกรรมที่สามารถติดตามได้อย่างชัดเจน อย่างเช่น วิ่ง, ว่ายน้ำ, ปั่นจักรยาน และมีปุ่มบวก (+) เพื่อเพิ่มกิจกรรมใหม่ที่เกิดขึ้น เช่น ผู้ใช้ต้องการบันทึกระยะทางและอัตราการเต้นของหัวใจขณะวิ่ง สามารถกดกิจกรรมเพื่อดูรายละเอียด และ เป็นต้น สามารถกดรูปโปรไฟล์เพื่อเข้าสู่หน้ารายละเอียดของผู้ใช้ที่แสดงข้อมูลโดยเฉลี่ยของการออกกำลังกาย

UI: การใช้สีที่แตกต่างกันสำหรับแต่ละกิจกรรม ช่วยให้ผู้ใช้แยกแยะได้ง่าย และการจัดวางข้อมูลในรูปแบบ List View ทำให้ดูเป็นระเบียบ

หน้าจอรายละเอียดกิจกรรมต่างๆ

UX: แสดงข้อมูลสรุปของแต่ละกิจกรรมที่บันทึกไว้ เช่น ระยะทาง, อัตราการเต้นของหัวใจ พร้อมปุ่มกลับไปหน้า Home

UI: การใช้กราฟิกแสดงข้อมูล เช่น แสดงตัวเลขของข้อมูลทำให้เข้าใจได้ง่ายขึ้น และการแบ่งส่วนของข้อมูลแต่ละประเภทออกจากกัน ช่วยให้ดูไม่รกตา

หน้าจอข้อมูลผู้ใช้งาน

UX: แสดงข้อมูลของผู้ใช้ที่บันทึกไว้ เช่น ชื่อผู้ใช้งาน ผลสรุปการออกกำลังกาย พร้อมปุ่มกลับไปหน้า Home

UI: การใช้กราฟิกแสดงข้อมูล เช่น แสดงข้อความและตัวเลขของข้อมูลทำให้เข้าใจได้ง่ายขึ้น พร้อมการแบ่งส่วนของข้อมูลแต่ละประเภทออกจากกัน ช่วยให้ดูไม่รกตา

d. Testing Design (5 คะแนน, -0.25)

- Unit Testing ทดสอบการทำงาน UI Components และการจัดการข้อมูลใน Local Database และการเรียกใช้ Repository เช่น การบันทึกข้อมูลการออกกำลังกาย การแสดงประวัติ และการตรวจสอบความถูกต้องของข้อมูลที่กรอก

- End-to-End (E2E) Testing ทดสอบการทำงานของแอปพลิเคชันตั้งแต่การกรอกข้อมูลจนถึงการแสดงผล เพื่อให้แน่ใจว่าผู้ใช้สามารถใช้งานฟีเจอร์ต่าง ๆ ได้อย่างสมบูรณ์

- Integration Testing ทดสอบการทำงานร่วมกันระหว่างส่วนต่าง ๆ ของแอป เช่น การส่งข้อมูลจาก UI ไปยัง Data Layer และการเรียกใช้ API

- Performance Testing ทดสอบความเร็วในการตอบสนองเมื่อเรียกใช้ API หรือดึงข้อมูลจากฐานข้อมูล

- Security Testing ทดสอบความปลอดภัยของข้อมูล เช่น การเข้ารหัสข้อมูลและการป้องกันการเข้าถึงโดยไม่ได้รับอนุญาต