ชื่อ.....

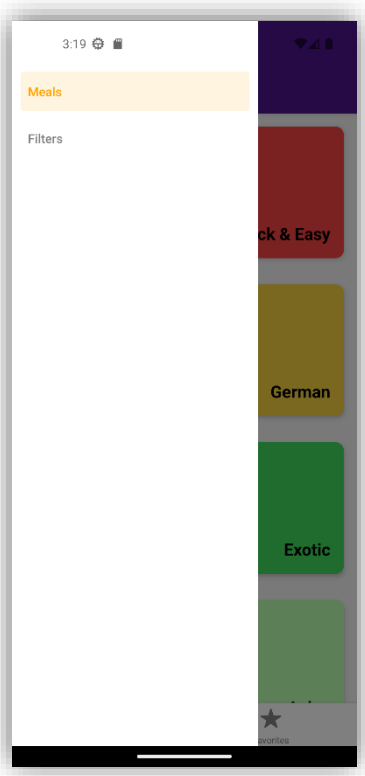
รหัสนักศึกษา.....

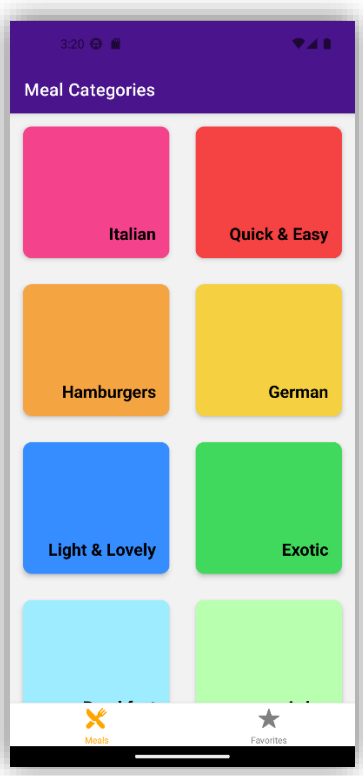
Lab 6 : Navigation (Part 2)

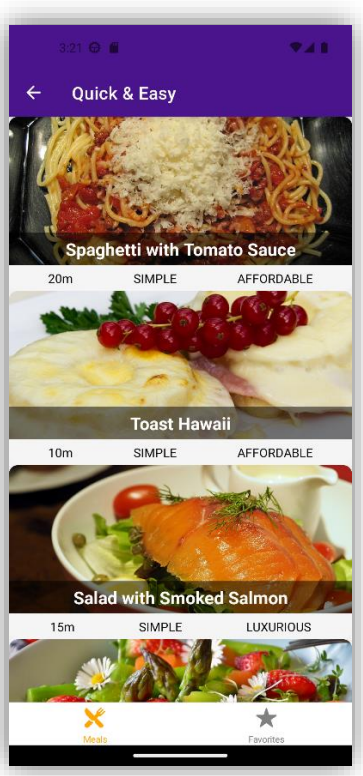
จงเขียนโปรแกรม MealApp ที่เป็นแอปพลิเคชันที่แสดงเมนูอาหารประเภทต่างๆ โดยได้มีการเพิ่มเติมจาก Lab 5 : Navigation (Part 1) ดังนี้

- สร้างแถบเมนูด้านข้าง แสดงเมนู Meals และ Filters
  - กรณีเลือกเมนู Meals แอปพลิเคชันจะแสดงหน้าต่าง Meal Categories และสร้าง tab ด้านล่าง แสดงแท็บเมนู Meals และ Favorites
    - กรณีเลือกแท็บเมนู Meals แอปพลิเคชันแสดงหน้าต่าง Meals Categories และผู้ใช้สามารถเลือกดูรายละเอียดเมนูอาหารต่างๆ ได้ (เหมือน Lab 5)
    - กรณีเลือกแท็บเมนู Favorites แอปพลิเคชันแสดงหน้าต่าง Your Favorites ซึ่งแสดงรายการอาหารที่ผู้ใช้ชอบ และผู้ใช้สามารถเลือกดูรายละเอียดเมนูอาหารต่างๆ ได้ (เหมือน Lab 5) โดยให้กำหนดรายการอาหารที่ชอบในโค้ดโปรแกรมโดยตรงเลย (ไม่สามารถเพิ่มหรือลบรายการอาหารที่ชอบได้)
  - กรณีเลือกเมนู Filters แอปพลิเคชันจะแสดงหน้าต่าง Filter Meals (ใช้แสดงผลเท่านั้น ยังไม่ได้เก็บข้อมูลฟิลเตอร์ และยังไม่สามารถใช้กรองข้อมูลจริงได้)

ตัวอย่างหน้าจอ (เลือกเมนูด้านข้าง Meals และเลือกแท็บเมนู Meals)



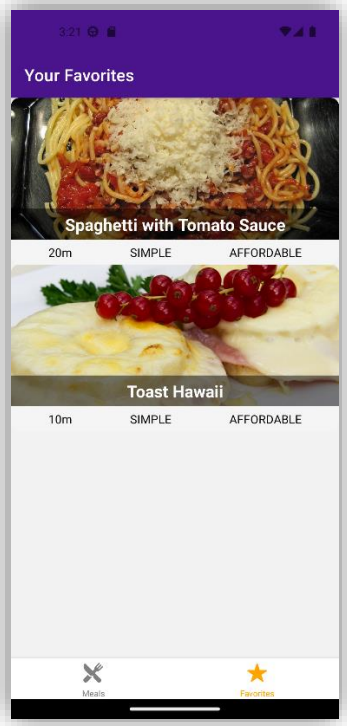


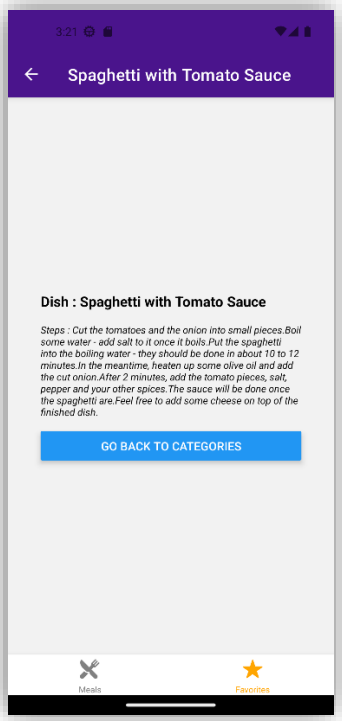


ชื่อ.....

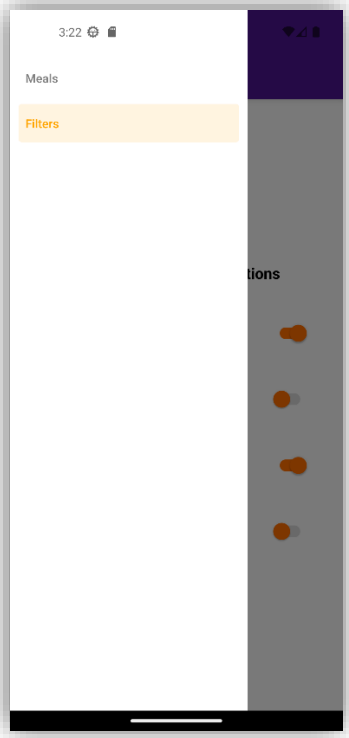
รหัสนักศึกษา.....

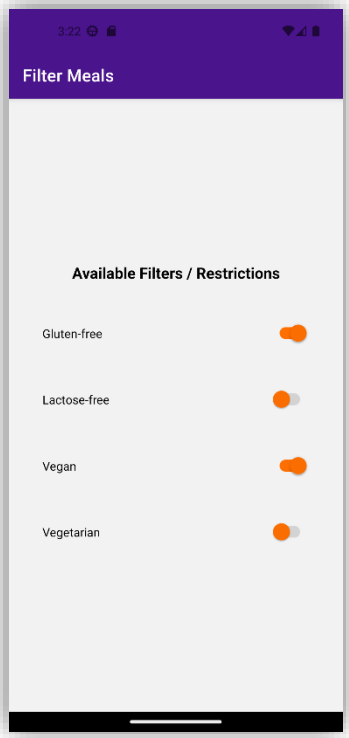
เลือกเมนูด้านข้าง Meals และเลือกแท็บเมนู Favorites





เลือกเมนูด้านข้าง Filters





ชื่อ.....

รหัสนักศึกษา.....

โครงสร้างของโปรแกรม

- **components**
  - CategoryGridTile.js
  - MealItem.js
  - MealList.js
- **data**
  - dummy-data.js
- **models**
  - category.js
  - meals.js
- **navigation**
  - MyNavigator.js
- **screens**
  - CategoriesScreen.js
  - CategoryMealsScreen.js
  - FavoritesScreen.js
  - FiltersScreen.js
  - MealDetailScreen.js
- App.js

หมายเหตุ นักศึกษาสามารถแก้ไขโค้ดโปรแกรมที่ได้เตรียมไว้ให้ (OnLearn) ได้ตามความเหมาะสม

ขั้นตอนการปฏิบัติการ

1. ให้นำโปรเจคที่ทำใน Lab 5 : Navigation (Part 1) มาปรับปรุง
2. ติดตั้ง library และ dependencies เพื่อใช้ในการทำ Navigation ให้ครบถ้วน
   - @react-navigation/bottom-tabs
   - @react-navigation/drawer
   - react-native-gesture-handler
   - react-native-reanimated

3. นศ.ต้องทำการสร้าง Nesting navigator ซึ่งประกอบด้วย Navigator ต่างๆ ดังนี้ (รายละเอียดการสร้างจะแสดงในลำดับต่อไป) (Slide 26-29)
   - **MealNavigator** เป็น stack navigator ที่ทำการเปลี่ยนหน้าจอระหว่างหน้า CategoriesScreen หน้า CategoryMealsScreen และหน้า MealDetailScreen (นศ.ทำไว้แล้วใน Lab 5 แต่อาจต้องปรับโค้ดจาก App.js ให้มาเขียนใน MyNavigator.js แทน)
   - **FavNavigator** เป็น stack navigator ที่ทำการเปลี่ยนหน้าจอระหว่างหน้า FavoritesScreen และหน้า MealDetailScreen
   - **MealsFavTabNavigator** เป็น bottom tab navigator ที่ทำการเปลี่ยน tab ระหว่างแท็บ Meals (เรียก **MealNavigator**) และ แท็บ Favorites (เรียก **FavNavigator**) (Slide 10)
   - **FiltersNavigator** เป็น stack navigator ที่จัดการการเปลี่ยนมายังหน้า FiltersScreen
   - **MainNavigator** เป็น drawer navigator ซึ่งเป็น navigator หลักของโปรแกรม ทำหน้าที่จัดการการเลือกเมนูระหว่างเมนู MealsFav (เรียก **MealsFavTabNavigator**) และเมนู Filters (เรียก **FiltersNavigator**) (Slide 21)

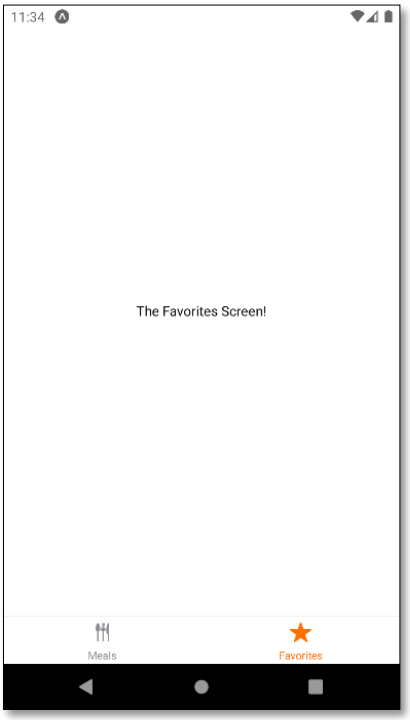
   หมายเหตุ รายละเอียดการสร้าง Nesting navigator แสดงในส่วนต่อไป หากนศ.เข้าใจโครงสร้างการทำงานแล้ว อาจไม่จำเป็นต้องดูรายละเอียดส่วนที่เหลือ แต่จะมีส่วนของข้อ 6-8 จะแสดงรายละเอียดการปรับการแสดงผลที่นอกเหนือจากการจัดการ Navigator เพิ่มเติม

4. **ปรับปรุงไฟล์ MyNavigator.js** เพื่อสร้างและกำหนด MealsFavTabNavigator ดังนี้
   - ทำการย้ายโค้ดส่วนของ MealsNavigator มาใช้ในไฟล์ MyNavigator.js (สร้างฟังก์ชันมา return MealNavigator) (Slide 26)
   - Import createBottomTabNavigator เพื่อใช้ในการสร้าง tab navigator
   - Import FavoritesScreen
   - ให้ทำการสร้าง MealsFavTabNavigator โดยเรียก createBottomTabNavigator() โดยกำหนด
     - Route name : Meals
       - component เป็นการอ้างอิง MealsNavigator
       - options ให้กำหนด property tabBarIcon ให้แสดงไอคอน ios-restaurant และกำหนดขนาดและสีตามชอบ (Slide 18)
     - Route name : Favorites
       - component เป็น FavoritesScreen

- options ให้กำหนด property tabBarIcon ให้แสดงไอคอน ios-star และกำหนดขนาดและสีตามชอบ

- ให้ปรับแต่งสีของข้อความและไอคอนบนแท็บ กรณีที่มีการเลือกแท็บ ใช้สีตามชอบ
- ทดลองรันและศึกษาการทำงานของโปรแกรม โดยโปรแกรมจะแสดงส่วน Tab ด้านล่าง ที่เมื่อกด Meals จะแสดงหน้าแรกของ MealsNavigator (CategoriesScreen) และเมื่อกด Favorites จะไปยังหน้า FavoritesScreen

  หมายเหตุ ในการทดลองรันการเรียก Navigator ในแต่ละขั้นตอน นศ.อาจต้องกำหนด navigator หลักในขณะนั้น (ในที่นี้คือ MealsFavTabNavigator) ไว้ในฟังก์ชัน MyNavigator() แล้วให้ App.js เรียก MyNavigator อีกที



5. **ปรับปรุงไฟล์ MyNavigator.js** เมื่อกด tab Favorites โปรแกรมควรจะแสดงรายการเมนูอาหารที่ผู้ใช้ชื่นชอบ และสามารถกดดูรายละเอียดแต่ละเมนูอาหารได้ ซึ่งมีการเปลี่ยนหน้าในรูปแบบ Stack ให้นศ.สร้างและกำหนด FavNavigator ดังนี้

   - ให้ทำการสร้าง FavNavigator เป็น stack navigator โดยกำหนด
     - Route name : Favorites
       - component เป็น FavoritesScreen
     - Route name : MealDetail

- component เป็น MealDetailScreen

- ปรับแต่งการแสดงผลตามชอบ
- ใน MealsFavTabNavigator แก้ไขโค้ดในส่วนของ Route name : Favorites ซึ่งเดิมกำหนดให้ component คือหน้า FavoritesScreen ให้กำหนด component เป็น FavNavigator

6. ปรับปรุงไฟล์ MealList.js

ไฟล์ MealList.js มีจุดประสงค์ให้เป็นการแสดงรายการเมนูอาหาร ซึ่งสามารถถูกเรียกได้ทั้งจาก CategorieMealsScreen (แสดงรายการเมนูอาหารตามประเภทอาหารที่เลือก) และ FavoritesScreen (แสดงรายการเมนูอาหารที่ผู้ใช้ชื่นชอบ) จึงควรมีการแยกคอมโพเนนต์ในส่วนแสดงรายการเมนูอาหารออกมา (MealList) และรับข้อมูลรายการอาหารที่ต้องการแสดงจาก CategoryMealsScreen และ FavoritesScreen อีกที

1. ในส่วนของ return() : ให้นศ.ทำการแสดงรายการเมนูอาหาร (Hint: นศ.สามารถคัดลอกโค้ดโปรแกรมจาก CategoryMealsScreen แล้วนำมาปรับปรุง) โดยกำหนดให้ข้อมูลที่จะแสดงใน FlatList เป็นข้อมูลที่ส่งผ่าน property ชื่อ listData
2. นศ.อาจต้องเพิ่มส่วนของฟังก์ชันสำหรับ property renderItem ใน FlatList รวมถึง StyleSheet ที่จำเป็นต้องใช้

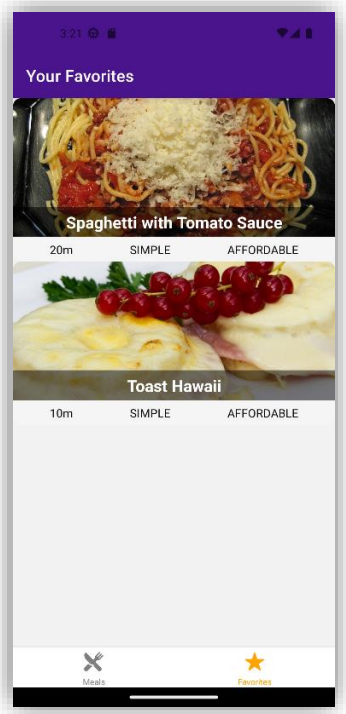
7. ปรับปรุงไฟล์ CategoryMealsScreen.js

1. import MealList มาใช้งาน
2. ในส่วนของ return() : ให้ทำการเรียกคอมโพเนนต์ MealList แทนที่จะสร้าง FlatList เพื่อแสดงรายการเมนูอาหารเอง โดยกำหนดให้ส่ง property ต่างๆ ดังนี้
   - listData : ลิสต์ข้อมูลเมนูอาหารที่ต้องการแสดง
   - navigation : {navigation} (เนื่องจาก MealList ไม่ได้ถูกกำหนดในการสร้าง Navigator จึงไม่มี property navigation ทำให้ CategoryMealsScreen ต้องส่ง property navigation ไปให้ MealList)
3. ลบหรือคอมเม้นต์โปรแกรมในส่วนที่ไม่จำเป็นต้องใช้ออก (ส่วนที่นำไปใช้ใน MealList แล้ว)
4. ทดลองรันโปรแกรม โปรแกรมจะสามารถทำงานได้ปกติ

8. ปรับปรุงไฟล์ FavoritesScreen.js

1. Import MealList และ {MEALS} มาใช้งาน

2. ทดลองสร้างตัวแปร favMeals กำหนดข้อมูลเมนูอาหารที่ชื่นชอบไว้ก่อน เพื่อใช้ทดสอบการทำงานของโปรแกรม
    - const favMeals = MEALS.filter((meal) => meal.id === "m1" || meal.id === "m2");
    - เป็นการกำหนดให้เมนูอาหารที่มี id เป็น ‘m1’ หรือ ‘m2’ เป็นรายการที่ชื่นชอบ
3. ในส่วนของ return() : ให้ทำการเรียกคอมโพเนนต์ MealList เพื่อแสดงรายการเมนูอาหารเอง โดยให้แสดงเมนูตามที่กำหนดใน favMeals (คล้ายกับหัวข้อปรับปรุงไฟล์ CategoryMealsScreen.js)
4. ทดลองรันโปรแกรม จะพบว่าเมื่อกด tab Favorites จะแสดงเมนูอาหารตามที่กำหนดใน favMeals ซึ่งเมื่อเข้าไปในเมนูหนึ่งๆ ก็แสดงหน้ารายละเอียดของเมนูนั้นต่อ



Drawer Navigation

9. ปรับปรุงไฟล์ MyNavigator.js

1. Import createDrawerNavigator เพื่อใช้ในการสร้าง drawer navigator
2. ให้ทำการสร้าง FiltersNavigator ซึ่งเป็น stack navigator โดยกำหนด
    - Route name : Filters
        - component เป็น FiltersScreen
3. ให้ทำการสร้าง MainNavigator ซึ่งเป็น drawer navigator โดยกำหนด

- Route name : MealsFav
    - component เป็น MealsFavTabNavigator
- Route name : Filters
    - component เป็น FiltersNavigator

4. เมื่อทดลองรัน นศ.จะพบว่าโปรแกรมจะเปิด drawer จากข้างจอ (Navigator หลัก คือ MainNavigator)

10. ปรับปรุงไฟล์ MyNavigator.js

1. ในส่วนของ MainNavigator ให้ทำการกำหนด drawerLabel ของ MealsFav เป็น “Meals”
2. นศ.สามารถตั้งค่าการแสดงผลของ drawer navigator ได้ตามชอบ (คล้ายตัวอย่างที่แสดง)
3. ทดลองรันโปรแกรม จะพบว่าส่วนของแถบเมนู จะแสดงหัวข้อ Meals และ Filters ซึ่งหัวข้อที่ทำงานอยู่จะมีสีตามที่กำหนด