

CS364 การพัฒนาโปรแกรมประยุกต์สำหรับอุปกรณ์พกพา

Hydro Tracker

จัดทำโดย

นาย กษิดิศ คงประพันธ์ 6609650129

นาย ภูตชญา กลินเดช 6609650160

นาย จันทร์พงศ์ วิทยอรุณรานี 6609650160

นาย ศุภกฤต ธรรมางกูร 6609650657

เสนอ

อาจารย์ ปกรณ์ ลีสุทธิพิรชัย

Hydro Tracker

Hydro Tracker แอปพลิเคชันที่ช่วยให้ผู้ใช้ทราบปริมาณน้ำที่ตนองค์ความดีมีในแต่ละวัน พร้อมพังก์ชันแจ้งเตือนการดื่มน้ำที่ผู้ใช้สามารถตั้งค่าเวลาการแจ้งเตือนได้ด้วยตนเอง อีกทั้งยังมีการเก็บประวัติการดื่มน้ำรายวัน เพื่อให้ผู้ใช้สามารถติดตามการดื่มน้ำของตนเองอย่างต่อเนื่อง

Logo ของแอปพลิเคชัน



หน้าหลักของแอปพลิเคชัน

หน้าหลักจะแสดงปริมาณน้ำที่ผู้ใช้ควรดื่มในแต่ละวัน โดยผู้ใช้จำเป็นต้องกรอกข้อมูล เช่น น้ำหนัก และ ระดับกิจกรรม



- กล่องแสดงผล (กล่องใหญ่) : แสดงผลข้อมูลของปริมาณน้ำที่ผู้ใช้ควรดื่มในแต่ละวัน สามารถแสดงข้อมูลแบบเรียลไทม์ ถึงแม้ผู้ใช้จะยังไม่ได้กดปุ่ม “คำนวนปริมาณน้ำ”
- ช่องกรอกน้ำหนัก : ให้ผู้กรอกน้ำหนักของตนเองเป็นเลขจำนวนเต็มลงไป โดยใช้เป็นหน่วย กิโลกรัม
- ช่องกรอกกิจกรรม : ให้ผู้ใช้เลือกระดับกิจกรรมโดยจะมี Dialog ขึ้นมาเพื่อให้ผู้ใช้เลือกระดับกิจกรรม 4 ระดับ เช่น ไม่ออกกำลังกาย, ออกกำลังกายเล็กน้อย, ออกกำลังกายปกติ, และออกกำลังกายหนัก
- ปุ่มคำนวนปริมาณน้ำ : เมื่อกดปุ่มนี้ จะพาผู้ใช้ไปยังหน้าเป้าหมาย
- เมนูนำทาง (ด้านล่าง) : มีเมนูที่จะพาผู้ใช้ไปยังหน้าต่างๆ เช่น หน้าหลัก, หน้าเป้าหมาย, และหน้าประวัติ

คุณสมบัติของแอปพลิเคชัน

1. หน้าหลัก : แสดงข้อมูลปริมาณน้ำที่ผู้ใช้ต้องดื่มให้ได้ในแต่ละวัน โดยใช้ข้อมูลน้ำหนักและระดับกิจกรรมในการคำนวน
2. หน้าเป้าหมาย : แสดงข้อมูลปริมาณน้ำที่ผู้ใช้ดื่มในวันนี้ เปรียบเทียบกับ ปริมาณน้ำเป้าหมายที่ต้องดื่มและมี Progress Bar เพื่อให้ผู้ใช้สามารถเห็นได้やすく สามารถเพิ่มปริมาณน้ำโดยการกดปุ่มปริมาณน้ำที่ต้องการเพิ่มหรือสามารถกรอกปริมาณน้ำที่ตนเองต้องการเพิ่มได้ สามารถตั้งค่าการตั้งเตือนได้โดยเลือกเวลาการแจ้งเตือนให้แจ้งเตือนทุกๆ กี่นาที
3. หน้าประวัติ : แสดงรายการการดื่มน้ำในแต่ละช่วงเวลา พร้อมทั้งปริมาณน้ำที่ดื่ม สามารถลบรายการการดื่มน้ำได้กรณีที่ผู้ใช้ใส่ข้อมูลผิด

การทำงานของแอปพลิเคชัน

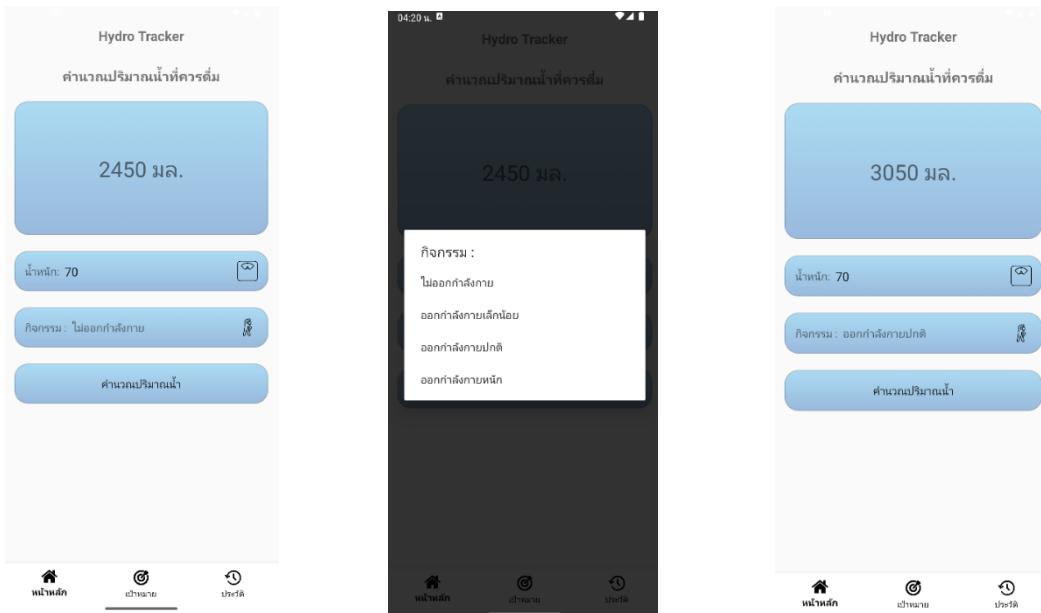
1. เริ่มต้นใช้งาน

ผู้ใช้เปิดแอปพลิเคชัน “Hydro Tracker”



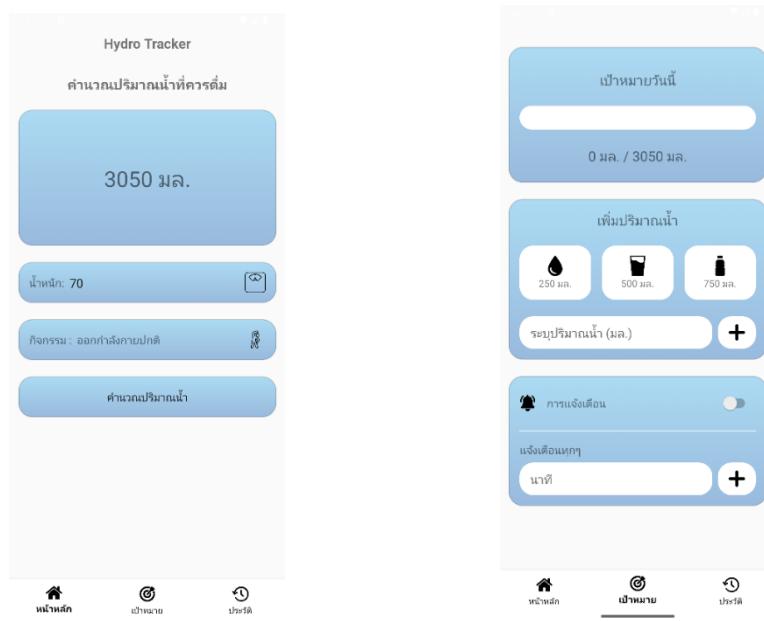
2. กรอกข้อมูลส่วนตัว และ แสดงผลปริมาณน้ำที่ควรบริโภค

ผู้ใช้กดกล่องน้ำหนักเพื่อกรอกน้ำหนักของตนเอง และ กดกล่องกิจกรรมโดยจะมี Dialog ขึ้นมาเพื่อให้เลือกระดับกิจกรรมของตนเอง จากนั้นระบบจะแสดงผลปริมาณน้ำที่ควรดื่ม



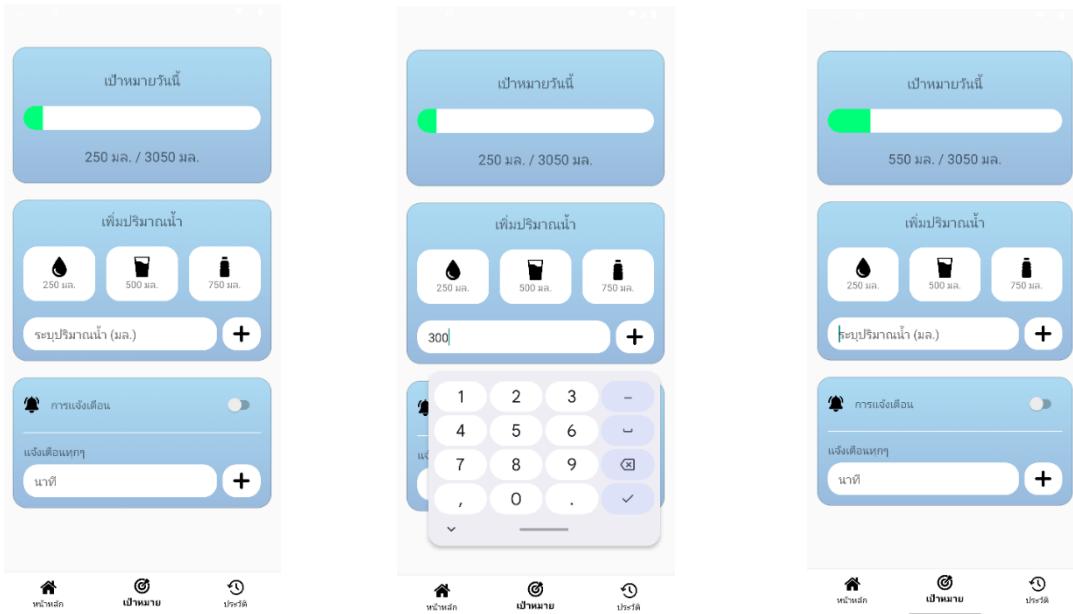
3. กดปุ่ม “คำนวณปริมาณน้ำ” และ ไปยัง “หน้าเป้าหมาย”

เมื่อกดปุ่มคำนวณปริมาณน้ำแล้วมายังหน้าเป้าหมาย ที่ปล่องเป้าหมายวันนี้จะแสดง Progress Bar และ ปริมาณน้ำที่ต้องไปแล้ว เปรียบเทียบกับ ปริมาณน้ำที่ควรต้อง



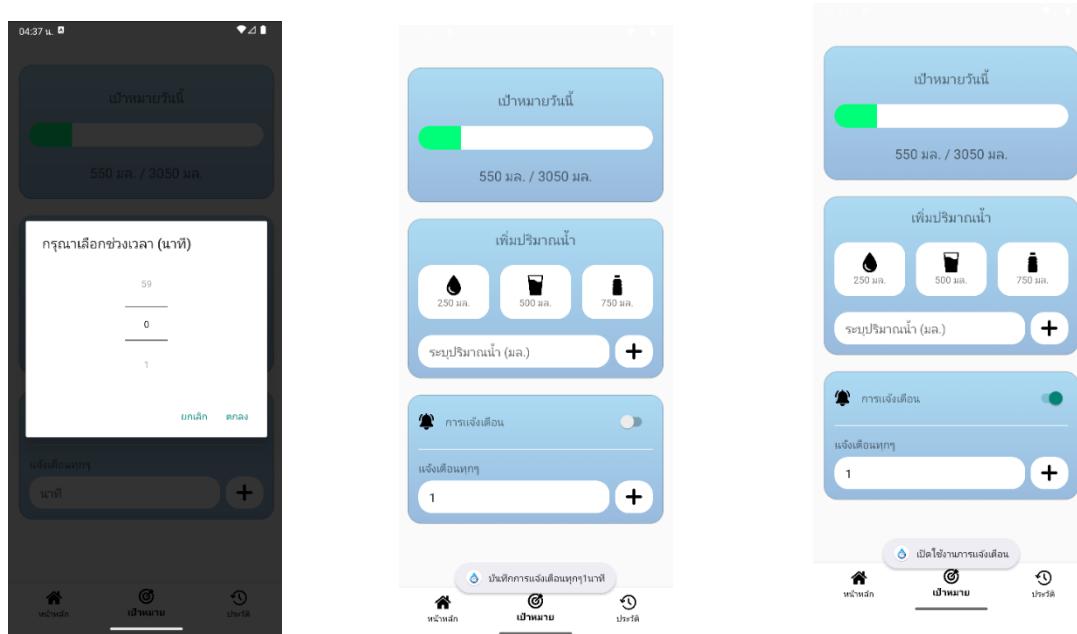
4. หน้าเป้าหมายที่ช่อง “เพิ่มปริมาณน้ำ”

ผู้ใช้สามารถเพิ่มปริมาณน้ำได้โดยการกดปุ่มไอคอนที่แอปพลิเคชันกำหนดไว้ให้ (250 มล. / 500 มล. / 750 มล.) หรือ กรอกปริมาณน้ำด้วยตนเองโดยกดปุ่มบางเพื่อเพิ่มรายการ



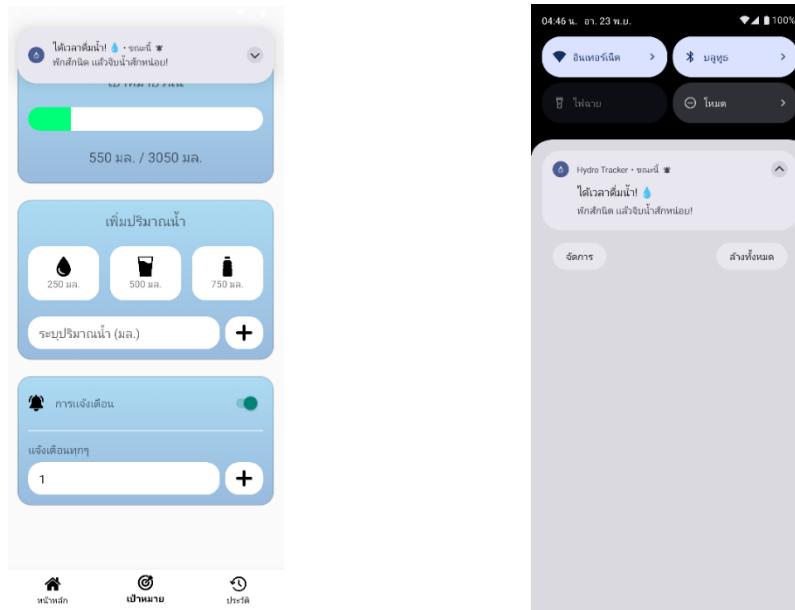
5. หน้าเป้าหมายที่ช่อง “การแจ้งเตือน”

ผู้ใช้สามารถเลือกเวลาที่ต้องการให้ระบบแจ้งเตือนการดื่มน้ำได้ในรูปแบบนาที โดยเมื่อกดเข้าไปจะมี Dialog เวลาขึ้นมา ให้เลือกเวลาที่ต้องการแจ้งเตือนทุกๆ กี่นาทีแล้วกด “ตกลง” หลังจากนั้นกด “ปุ่มบวก” จะมี Toast แสดงว่า “บันทึกการแจ้งเตือนทุกๆ x นาที” ตามที่เราได้กำหนดไว้ หลังจากนั้นกดที่ปุ่มเปิดการแจ้งเตือน (จากปุ่มสีเทา -> ปุ่มสีเขียว) จะมี Toast แสดงว่า “เปิดใช้งานการแจ้งเตือน”



6. การแจ้งเตือน

เมื่อครบกำหนดเวลาที่ผู้ใช้ตั้ง จะมีการแจ้งเตือนเด้งขึ้นมาบนสุดของแอปพลิเคชัน โดยเมื่อสไลเดอร์หน้าจอจะเห็นการแจ้งเตือนนั้นและเมื่อกดที่การแจ้งเตือนนั้นจะพาไปยังแอปพลิเคชัน



7. หน้าประวัติ

ระบบจะแสดงรายการที่ผู้ใช้ดิ่งน้ำ (แสดงรายการล่าสุดก่อน) โดยจะแสดงปริมาณน้ำที่ดิ่งเวลาที่ดิ่ง และสามารถลบรายการนั้นได้ เมื่อลบรายการนั้น ที่ “หน้าเป้าหมาย” ข้อมูลปริมาณน้ำที่ดิ่งไปแล้วจะลดลง และ Progress Bar ก็จะลดลงเช่นกัน



พังก์ชันการทำงานในส่วนต่างๆ

1. พังก์ชันหลัก (MainActivity)

The screenshot shows the Android Studio interface with the code editor open to MainActivity.java. The code implements a BottomNavigationView and initializes WaterTracker. The code editor has syntax highlighting and includes annotations for methods like onCreate.

```
1 package com.example.cs364_project;
2
3 import android.os.Bundle;
4
5 import androidx.appcompat.app.AppCompatActivity;
6 import androidx.fragment.app.Fragment;
7
8 import com.google.android.material.bottomnavigation.BottomNavigationView;
9
10 public class MainActivity extends AppCompatActivity implements HomeFragment.HomeFragmentListener {
11
12     private BottomNavigationView bottomNav;
13
14     private int lastTotalMl = 0;
15
16     @Override
17     protected void onCreate(Bundle savedInstanceState) {
18         super.onCreate(savedInstanceState);
19         setContentView(R.layout.activity_main);
20
21         // Initialize SharedPreferences -> WaterTracker
22         WaterTracker.init(getApplicationContext());
23     }
24 }
```

- โหลดข้อมูลเป้าหมายน้ำที่คราวดีมต่อวันจาก SharedPreferences เก็บไว้ในตัวแปร lastTotalML
- แสดง Fragment เมื่อเปิดแอปพลิเคชัน โดยจะแสดงหน้า HomeFragment เป็นหน้าแรก
- ควบคุมแถบ Bottom Navigation
- ส่งข้อมูลที่ผู้ใช้ป้อนแล้วคำนวนเป็นปริมาณน้ำที่ควรดื่มต่อวันเก็บไว้ใน TOTAL_ML และส่งข้อมูลไปที่ GoalFragment

2. พื้นฐานหน้าการใช้งาน (HomeFragment, GoalFragment, HistoryFragment)

The screenshot shows the Android Studio interface. On the left, the project structure is visible with packages like com.example.cs364.project containing files such as HomeActivity, AlarmReceiver, BootReceiver, GoalFragment, HistoryFragment, and WaterTracker. On the right, the HomeFragment.java file is open in the code editor. The code defines a HomeFragment class that extends Fragment. It includes imports for Context, View, and Fragment. The class contains several private member variables: tvWaterTotal, tvActivityValue, edtWeight, activityLevel, totalML, lastTotalML, and waterAnimator. The code also includes comments explaining the logic for calculating water intake based on activity level and weight.

```

package com.example.cs364_project;
import ...;
// น้ำเพลิดในการใช้อุปกรณ์น้ำหนัก + ร่องรอยการดื่มน้ำ -> คำนวณเป็นปริมาณน้ำที่ต้องดื่มต่อวัน
public class HomeFragment extends Fragment {
    private TextView tvWaterTotal;
    private TextView tvActivityValue;
    private EditText edtWeight;
    private String activityLevel = null;
    private int totalML = 0;
    private int lastTotalML = 0;
    private ValueAnimator waterAnimator;
}

```

หน้า HomeFragment :

- รับค่าจากผู้ใช้ เช่น น้ำหนัก และ ระดับกิจกรรม
- คำนวนปริมาณน้ำที่ควรดื่ม โดยการนำน้ำหนัก (กก.) \times 35 และบวกด้วยระดับกิจกรรม ไม่ออกกำลังกายจะบวกด้วยค่า 0, ออกกำลังกายเล็กน้อยจะบวกด้วยค่า 300, ออกกำลังกายปกติจะบวกด้วยค่า 600, ออกกำลังกายหนักจะบวกด้วยค่า 900
- ส่งผลการคำนวนไปที่ MainActivity

```

1 package com.example.cs364_project;
2
3 > import ...
4
5 // หน้าในการเพิ่มเป้าหมาย + ตั้งเวลาเพื่อเตือน + progress bar
6
7 <> public class GoalFragment extends Fragment {
8
9     5 usages
10    private ProgressBar progressBar;
11
12    3 usages
13    private TextView tvWaterTotal;
14
15    2 usages
16    private LinearLayout btnAdd250, btnAdd500, btnAdd750, btnAddCustom;
17
18    3 usages
19    private EditText edtCustomAmount;
20
21    5 usages
22    private Switch switchReminder;
23
24    14 usages
25    private EditText edtInterval;
26
27    3 usages
28    private LinearLayout btnSaveInterval;
29
30
31
32
33
34
35
36
37
38
39
40
41

```

หน้า GoalFragment :

- แสดง Progress Bar โดยแสดงแบบปริมาณน้ำที่ดื่มไปแล้วเปรียบเทียบกับปริมาณน้ำเป้าหมาย
- เพิ่มปริมาณน้ำที่ดื่ม เมื่อผู้ใช้กดปุ่ม 250 มล, 500 มล, 750 มล, ใส่จำนวนด้วยตนเอง
- ตั้งค่าการแจ้งเตือนการดื่มน้ำ ผู้ใช้สามารถเปิด/ปิดปุ่มการแจ้งเตือน สามารถเลือกระยะเวลาแจ้งเตือน (นาที) บันทึกเวลาแจ้งเตือน โดยระบบจะบันทึกข้อมูลลง SharedPreferences และตั้งค่าเวลาผ่าน ReminderScheduler
- มีการขอสิทธิ์ (permission) การแจ้งเตือนที่ฟังก์ชัน requestNotificationPermissionIfNeeded()

```

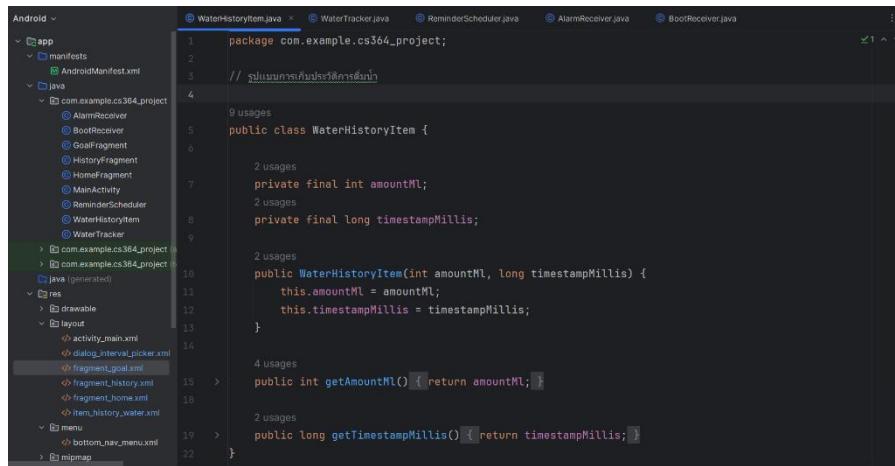
1 package com.example.cs364_project;
2
3 > import ...
4
5 // หน้าในการดูประวัติการดื่มน้ำทั้งหมด + สามารถซื้อมัน
6
7 <> public class HistoryFragment extends Fragment {
8
9     5 usages
10    private LinearLayout layoutHistoryList;
11
12    1 usage
13    private final SimpleDateFormat timeFormatter =
14        new SimpleDateFormat("HH:mm", Locale.getDefault());
15
16    public HistoryFragment() {}
17
18    @Nullable
19    @Override
20    public View onCreateView(@NonNull LayoutInflater inflater,
21                           @Nullable ViewGroup container,
22                           @Nullable Bundle savedInstanceState) {
23        return inflater.inflate(R.layout.fragment_history, container, false);
24    }
25
26
27
28
29
30
31
32
33
34
35
36
37

```

หน้า HistoryFragment :

- ดึงประวัติการดื่มน้ำจาก WaterTracker โดยใช้ WaterTracker.getHistory()
- แสดงรายการข้อมูลประกอบด้วย ปริมาณน้ำ, เวลา, บุํลบรรยายการ
- ลบรายการที่ไม่ต้องการโดยเรียกใช้ btnDelete และ Progress ในหน้า GoalFragment จะลดลงด้วย เพราะใช้ WaterTracker เดียวกัน แล้วเรียกใช้ renderHistory() เพื่ออัปเดตหน้า

3. พังก์ชันการเก็บข้อมูล (WaterHistory, WaterTracker)



```
Android
app
  manifests
    AndroidManifest.xml
  java
    com.example.cs364_project
      AlarmReceiver.java
      BootReceiver.java
      GoalFragment.java
      HistoryFragment.java
      HomeFragment.java
      MainActivity.java
      ReminderScheduler.java
      WaterHistoryItem.java
      WaterTracker.java
    java (generated)
    res
      drawable
      layout
        activity_main.xml
        dialog_interval_picker.xml
        fragment_goal.xml
        fragment_history.xml
        fragment_home.xml
        item_history_water.xml
      menu
        bottom_nav_menu.xml
      mipmap
```

```
WaterHistoryItem.java
package com.example.cs364_project;
// รับเมมการเก็บประวัติการดื่มน้ำ
public class WaterHistoryItem {
    private final int amountML;
    private final long timestampMillis;

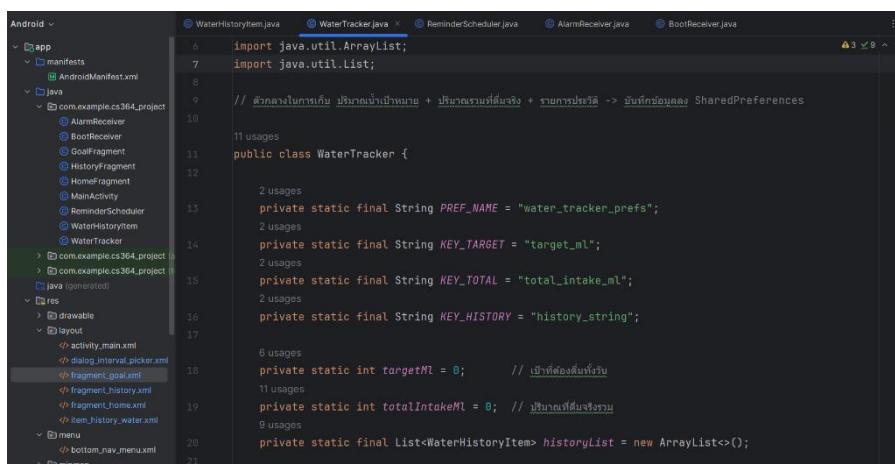
    public WaterHistoryItem(int amountML, long timestampMillis) {
        this.amountML = amountML;
        this.timestampMillis = timestampMillis;
    }

    public int getAmountML() { return amountML; }

    public long getTimestampMillis() { return timestampMillis; }
}
```

หน้า WaterHistoryItem :

- Object ที่เก็บปริมาณน้ำและเวลา



```
Android
app
  manifests
    AndroidManifest.xml
  java
    com.example.cs364_project
      AlarmReceiver.java
      BootReceiver.java
      GoalFragment.java
      HistoryFragment.java
      HomeFragment.java
      MainActivity.java
      ReminderScheduler.java
      WaterHistoryItem.java
      WaterTracker.java
    java (generated)
    res
      drawable
      layout
        activity_main.xml
        dialog_interval_picker.xml
        fragment_goal.xml
        fragment_history.xml
        fragment_home.xml
        item_history_water.xml
      menu
        bottom_nav_menu.xml
      mipmap
```

```
WaterTracker.java
import java.util.ArrayList;
import java.util.List;

// ผูกต่อไปยังคลาส WaterHistoryItem > ปริมาณรวมที่ดื่ม + สถานะปัจจุบัน -> บันทึกลงใน Shared Preferences
public class WaterTracker {
    private static final String PREF_NAME = "water_tracker_prefs";
    private static final String KEY_TARGET = "target_ml";
    private static final String KEY_TOTAL = "total_intake_ml";
    private static final String KEY_HISTORY = "history_string";

    private static int targetML = 0; // จำนวนที่ต้องดื่มน้ำ
    private static int totalIntakeML = 0; // จำนวนที่ดื่มจริงๆ
    private static final List<WaterHistoryItem> historyList = new ArrayList<>();
}
```

หน้า WaterTracker :

- โหลดข้อมูลเป้าหมาย, ปริมาณน้ำที่ดื่มไปแล้ว, ประวัติการดื่มน้ำจาก SharedPreferences
- เก็บเป้าหมายน้ำที่ต้องดื่ม โดยใช้ setTarget(), getTarget()
- บันทึกปริมาณน้ำที่ดื่ม addEntry(amountML) และลบประวัติ removeEntry()

4. พัฒนาการแจ้งเตือน (AlarmReceiver, BootReceiver, ReminderScheduler)

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `AlarmReceiver.java` file:

```
1 package com.example.cs364_project;
2
3 > import ...;
4
5 public class AlarmReceiver extends BroadcastReceiver {
6
7     2 usages
8     public static final String CHANNEL_ID = "water_reminder_channel";
9
10    1 usage
11    private static final int NOTIFICATION_ID = 1001;
12
13    @Override
14    public void onReceive(Context context, Intent intent) {
15
16        // 1) ปลุกหน้าจอ (ต่ำบล็อก)
17        wakeScreen(context);
18
19        // 2) สร้าง Intent ใหม่มาส่งไป GoalFragment
20        Intent openIntent = new Intent(context, MainActivity.class);
21
22        // เปิด goal fragment
23        openIntent.putExtra("name", "OPEN_FRAGMENT", "GOAL");
24
25        // นำ Intent ไปสู่ Activity ต่อไป
26    }
27}
```

หน้า AlarmReceiver :

- แสดง Notification แจ้งเตือนการดื่มน้ำโดยการสร้าง Notification พร้อมข้อความ แจ้งเตือนและเมื่อกดที่ Notification จะพาไปแอปพลิเคชัน
- ปลุกหน้าจอโดยใช้ `wakeScreen(Context)`

```

Android
app
  manifests
    AndroidManifest.xml
  java
    com.example.cs364_project
      AlarmReceiver
      BootReceiver
      GoalFragment
      HistoryFragment
      HomeFragment
      MainActivity
      ReminderScheduler
      WaterHistoryItem
      WaterTracker
    java (generated)
    res
      drawable
      layout
        activity_main.xml
        dialog_interval_picker.xml
        fragment_goal.xml
        fragment_history.xml
        fragment_home.xml
        item_history_water.xml
      menu
      bottom_nav_menu.xml
      mipmap

```

```

1 package com.example.cs364_project;
2
3 import android.content.BroadcastReceiver;
4 import android.content.Context;
5 import android.content.Intent;
6 import android.content.SharedPreferences;
7
8 </>
9
10 public class BootReceiver extends BroadcastReceiver {
11
12     @Override
13     public void onReceive(Context context, Intent intent) {
14         if (Intent.ACTION_BOOT_COMPLETED.equals(intent.getAction())) {
15
16             SharedPreferences prefs = context.getSharedPreferences(name: "reminder_prefs", Context.MODE_PRIVATE);
17             boolean isOn = prefs.getBoolean(key: "reminden_on", defValue: false);
18             long intervalMillis = prefs.getLong(key: "interval_millis", defValue: 0L);
19
20             if (isOn && intervalMillis > 0) {
21                 ReminderScheduler.scheduleRepeatingAlarm(context, intervalMillis);
22             }
23         }
24     }
25 }

```

หน้า BootReceiver :

- ทำงานทันทีหลังเปิดเครื่อง (BOOT เสร็จ) โดยเช็คว่าผู้ใช้เปิดแจ้งเตือนดีมั่น้ำใจหรือไม่ถ้าเปิดก็ตั้งแจ้งเตือนใหม่อีกครั้ง

```

Android
app
  manifests
    AndroidManifest.xml
  java
    com.example.cs364_project
      AlarmReceiver
      BootReceiver
      GoalFragment
      HistoryFragment
      HomeFragment
      MainActivity
      ReminderScheduler
      WaterHistoryItem
      WaterTracker
    java (generated)
    res
      drawable
      layout
        activity_main.xml
        dialog_interval_picker.xml
        fragment_goal.xml
        fragment_history.xml
        fragment_home.xml
        item_history_water.xml
      menu
      bottom_nav_menu.xml
      mipmap

```

```

1 package com.example.cs364_project;
2
3 > import ...
4
5 4 usages
6
7 public class ReminderScheduler {
8
9     1 usage
10     public static final int REQUEST_CODE_ALARM = 2001;
11
12     3 usages
13     public static void scheduleRepeatingAlarm(Context context, long intervalMillis) {
14
15         AlarmManager alarmManager = (AlarmManager) context.getSystemService(Context.ALARM_SERVICE);
16         if (alarmManager == null) return;
17
18         PendingIntent pendingIntent = buildAlarmPendingIntent(context);
19
20         long triggerAtMillis = SystemClock.elapsedRealtime() + intervalMillis;
21
22         // ELAPSED_REALTIME_WAKEUP + Repeating
23         alarmManager.setRepeating(
24             AlarmManager.ELAPSED_REALTIME_WAKEUP,
25             triggerAtMillis, SystemClock.elapsedRealtime() + intervalMillis,
26             pendingIntent);
27     }
28 }

```

หน้า ReminderScheduler :

- จัดการการตั้ง Alarm แจ้งเตือนดีมั่น้ำและยกเลิก Alarm
- มีการใช้ ELAPSED_REALTIME_WAKEUP เพื่อปลุกหน้าจอเมื่อเครื่องจะปิดอยู่
- ตั้ง Alarm ให้มีการแจ้งเตือนซ้ำๆ โดยใช้ฟังก์ชัน scheduleRepeatingAlarm()
- ยกเลิกการตั้ง Alarm โดยใช้ฟังก์ชัน cancelAlarm()

ประเด็นความท้าทายทางเทคนิค

1. การออกแบบแบบ Bottom Navigation ร่วมกับ Fragment

ในตอนแรกของการออกแบบได้มีการออกแบบแต่ละส่วนแบบใช้ LinearLayout แล้วนำ BottomNavigationView ไปใส่ส่วนล่างใน Layout ต่างๆ ปรากฏว่าเมื่อทำในรูปแบบนี้จะเกิดปัญหาที่เวลาในการเปลี่ยนหน้า Bottom Navigation จะมีการเปลี่ยนแปลงไปด้วย ซึ่งเราแก้ไขโดยการออกแบบให้หน้า activity_main.xml เป็นโครงที่มีส่วนของ FrameLayout แล้วก็ BottomNavigation และใช้ MainActivity.java ในการเรียกเปิดหน้า Fragment ต่างๆ เช่น HomeFragment.java, GoalFragment.java, HistoryFragment.java

2. การตั้งค่าการแจ้งเตือนผู้ใช้งาน

ในส่วนนี้จะมีความยากในการตั้งค่าขอสิทธิ์การอนุญาตต่างๆ (permission) โดยใน Android เวอร์ชันใหม่ต้องมีการขอ POST_NOTIFICATIONS จากผู้ใช้ก่อน การตั้งค่าจะไม่สามารถได้รับการแจ้งเตือนได้แม้ขาดส่วนใดส่วนหนึ่งทำให้ยากในการตรวจสอบ ซึ่งเราแก้ไขโดยการตั้งค่าในไฟล์ AndroidManifest.xml และใน GoalFragment.java ในส่วนของฟังก์ชัน requestNotificationPermissionIfNeeded()

3. การจัดการข้อมูลที่ต้องใช้ร่วมกันระหว่างหลาย Fragment ผ่าน WaterTracker

ถ้าไม่มีข้อมูลกลางที่ใช้ในการเก็บข้อมูลจะทำให้ข้อมูลในแต่ละ Fragment ไม่ตรงกัน เช่นหน้า GoalFragment กับ HistoryFragment เราแก้ปัญหาโดยการสร้าง Class WaterTracker เพื่อเป็นตัวกลางในการเก็บข้อมูล และบันทึกและโหลดข้อมูลจาก SharedPreferences

4. การเชื่อมต่อข้อมูลของ Progress Bar กับ ประวัติการดื่มน้ำ

ในตอนแรก UI ของ Progress Bar ไม่อัปเดตค่าถึงแม้ผู้ใช้จะเพิ่ม/ลบ รายการการดื่มน้ำโดยเราแก้ปัญหาให้เรียกใช้ข้อมูลจาก WaterTracker และทำการรีเฟรช UI โดยใช้ฟังก์ชัน refreshProgressUi()

รายชื่อสมาชิกและหน้าที่

นาย กษิดิศ คงประพันธ์

- ออกแบบ UI ของแอปพลิเคชัน, จัดการไฟล์เดอร์ drawable

นาย กฤตชญา กลินเดช

- ออกแบบ UI ของแอปพลิเคชัน และ จัดการไฟล์เดอร์ values, นำเสนองาน

นาย จันทร์พงศ์ วิทยอรุณรานี

- ทดสอบระบบ และ ตรวจสอบให้ตรงตามเงื่อนไข

นาย ศุภกฤต ธรรมางกูร

- จัดการไฟล์เดอร์ layout และ menu, เชื่อมต่อกับไฟล์ Java ต่างๆ, นำเสนอ

Link : Github ของโครงการ

<https://github.com/SupakitThammangoon/CS364-Project.git>

Link : Presentation

<https://youtu.be/bp7rpgmFVRw>