Embedded System Software
Final Project - ootdPT

Junhyeok Park
20171643

Motivation

IoT ***(Internet on Things)***

Motivation

Ex) Intercom

COMMAX SmartHome & Security

KOCOM

Motivation

What about

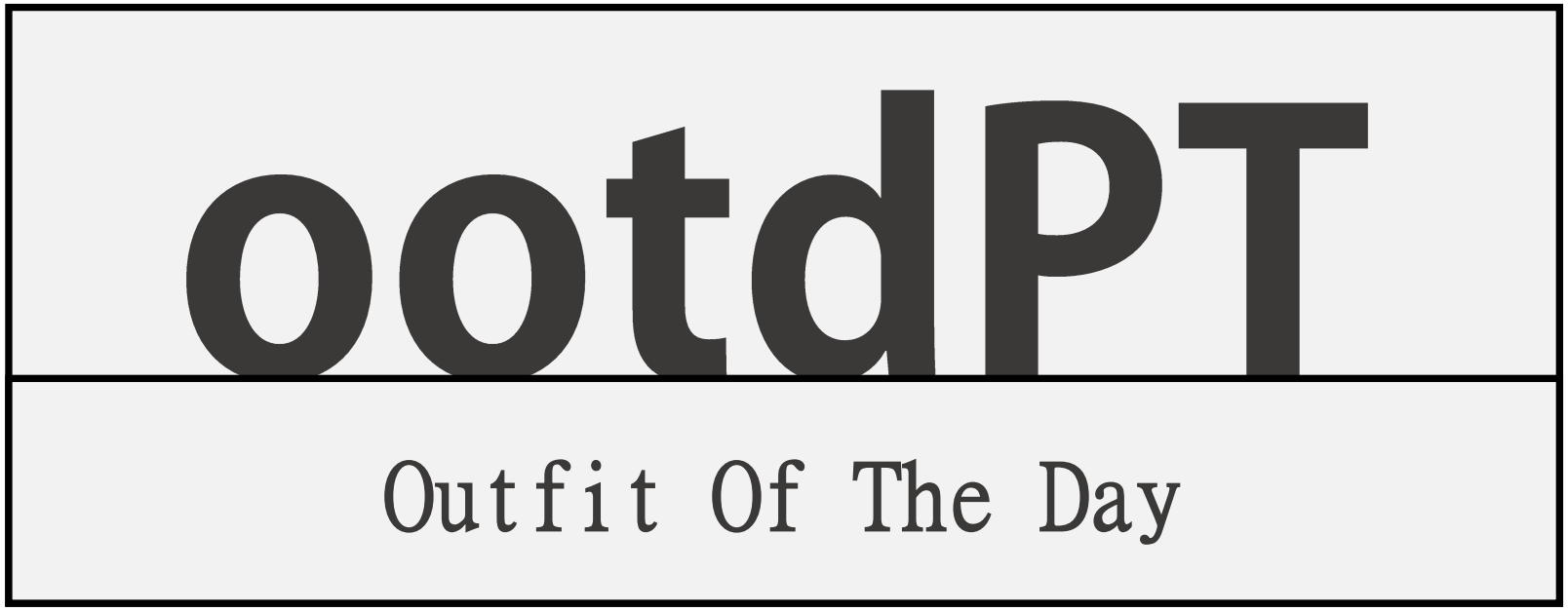
outfit recommendation?

Motivation

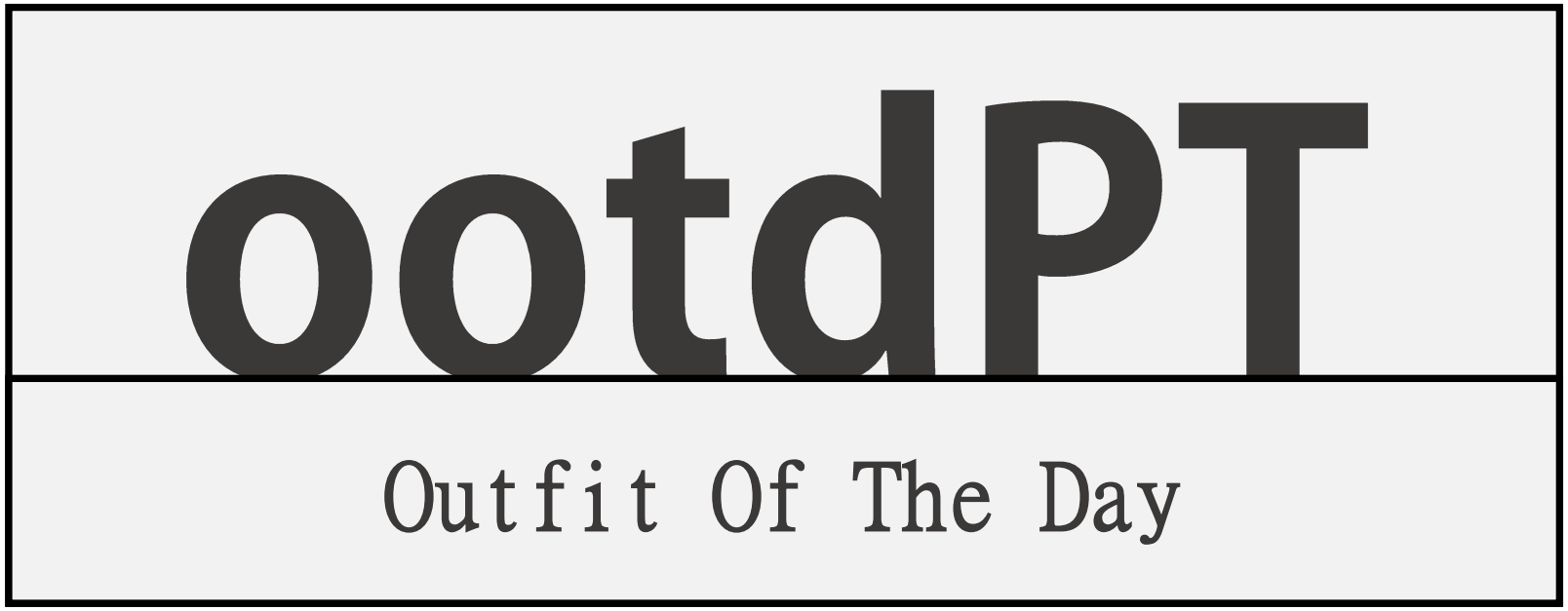
What about

outfit recommendation?

Idea

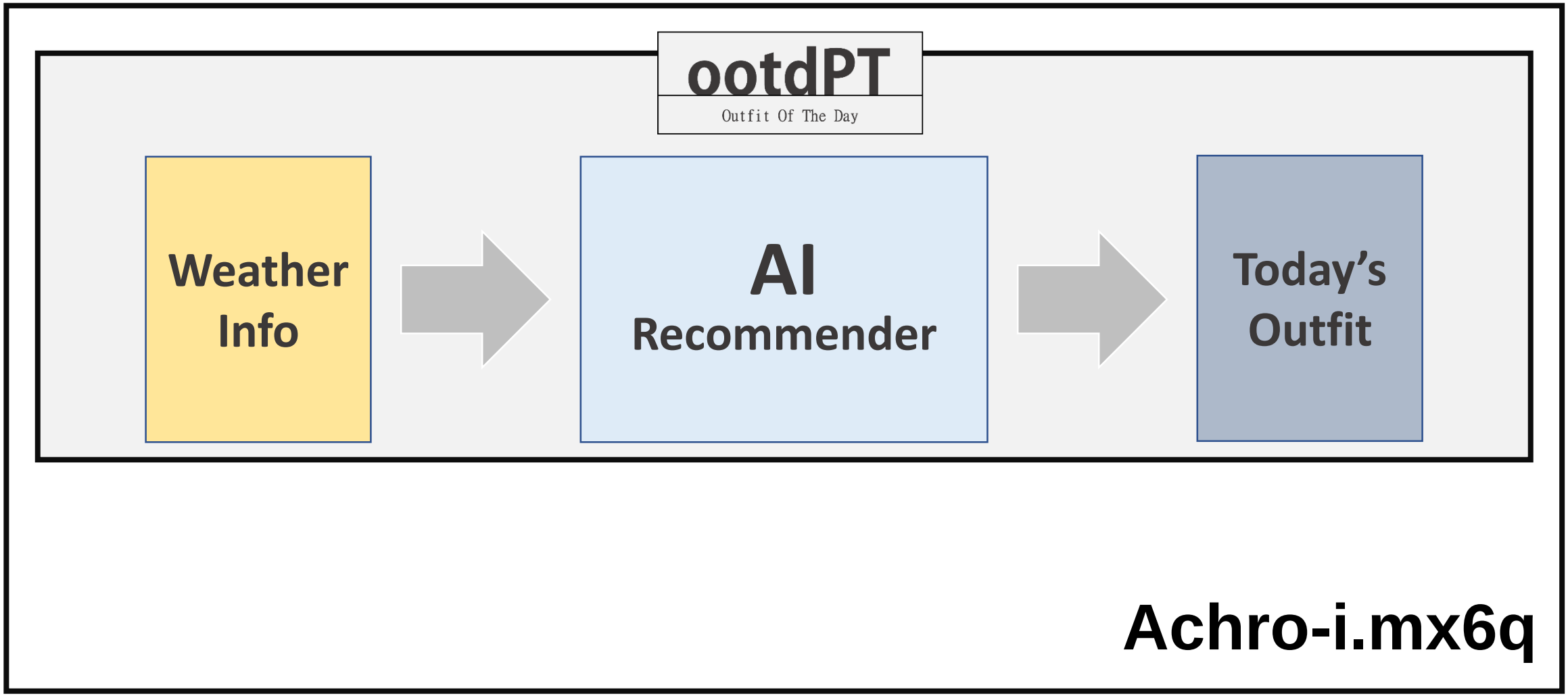


Idea

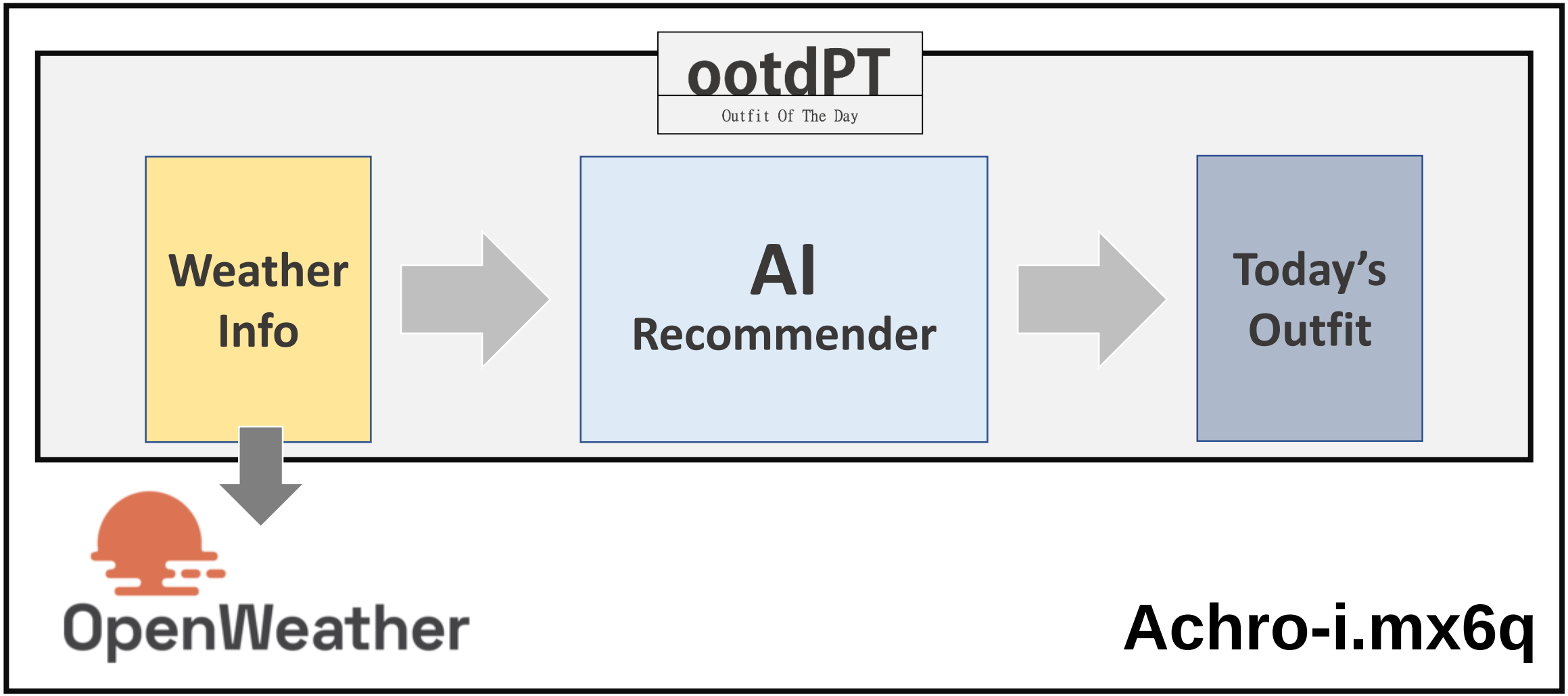


Android Application
on achro-i.mx6q device

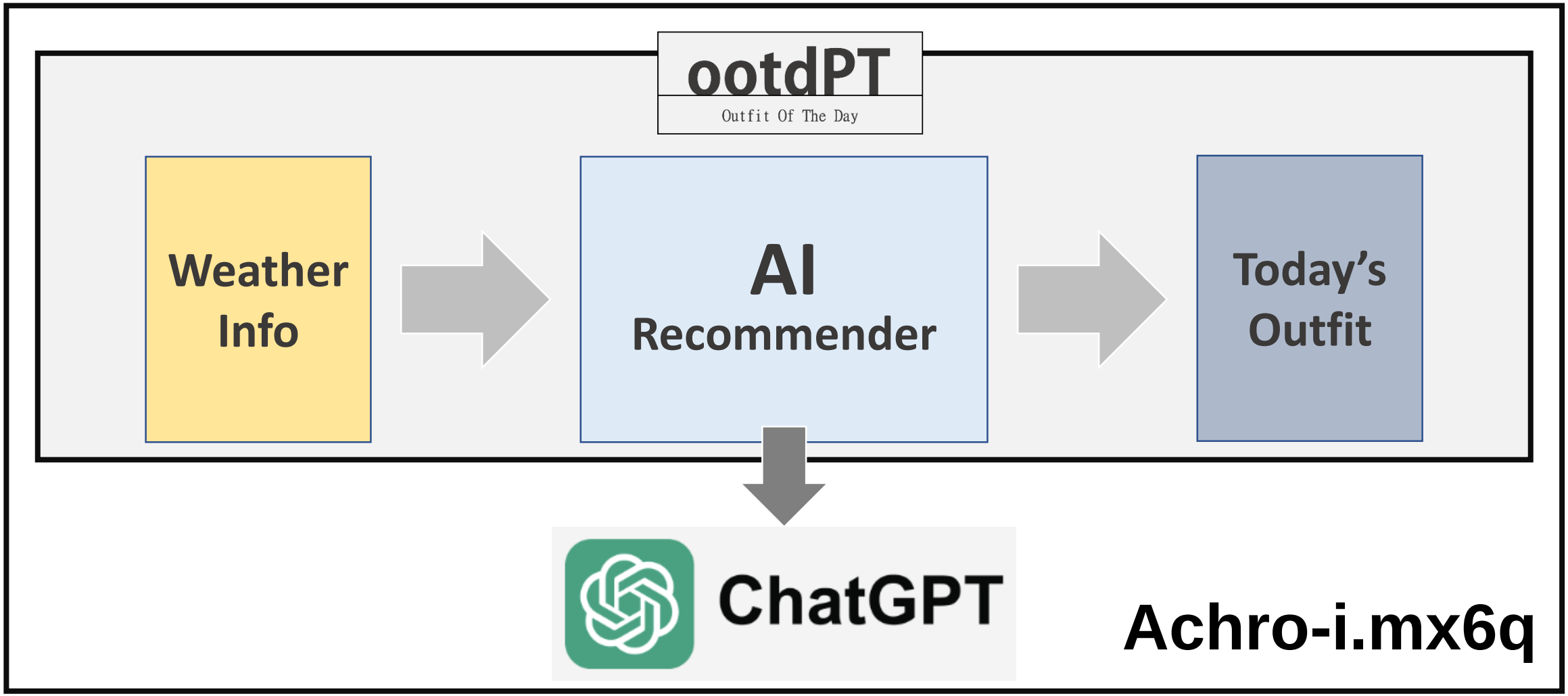
Idea


ootdPT
Outfit Of The Day
Weather Info
AI Recommender
Today's Outfit
Achro-i.mx6q

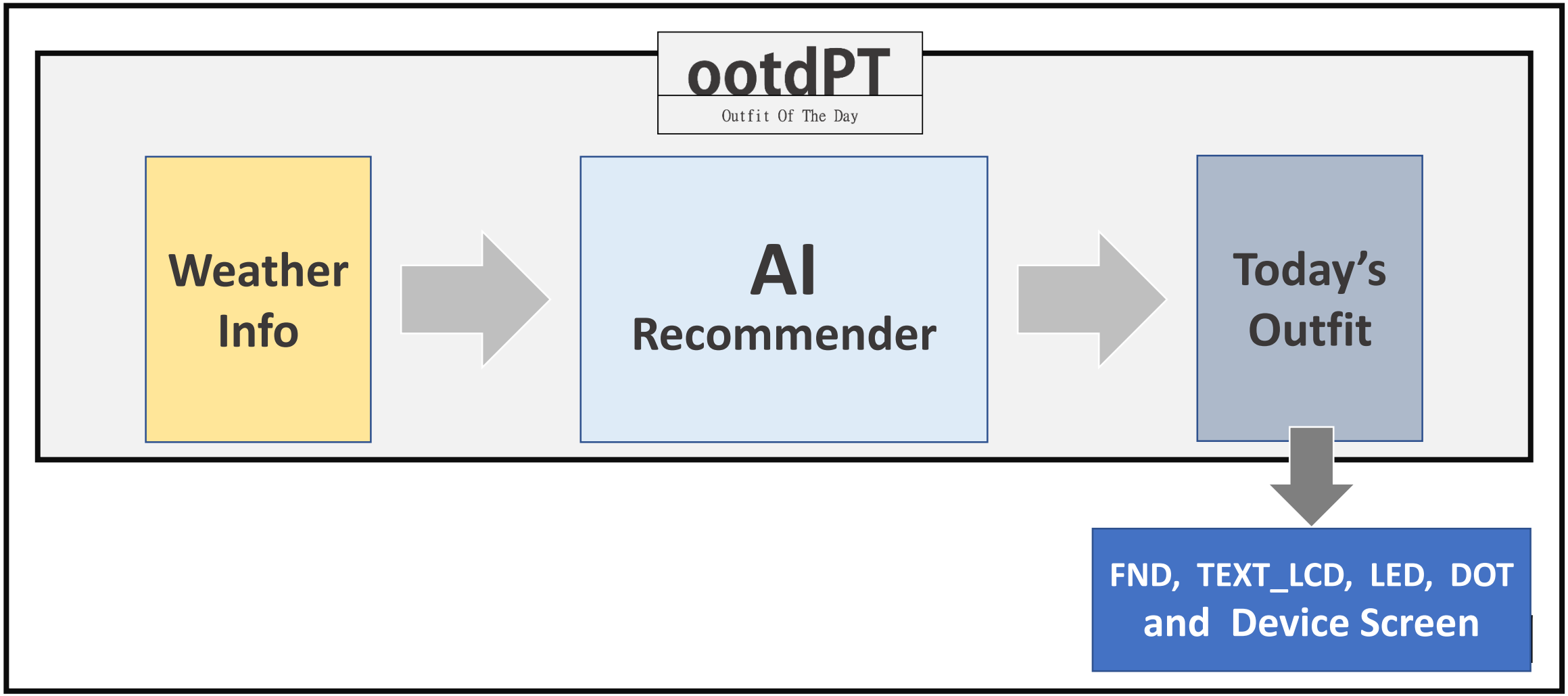
Idea


ootdPT
Outfit Of The Day
Weather Info
AI Recommender
Today's Outfit
OpenWeather
Achro-i.mx6q

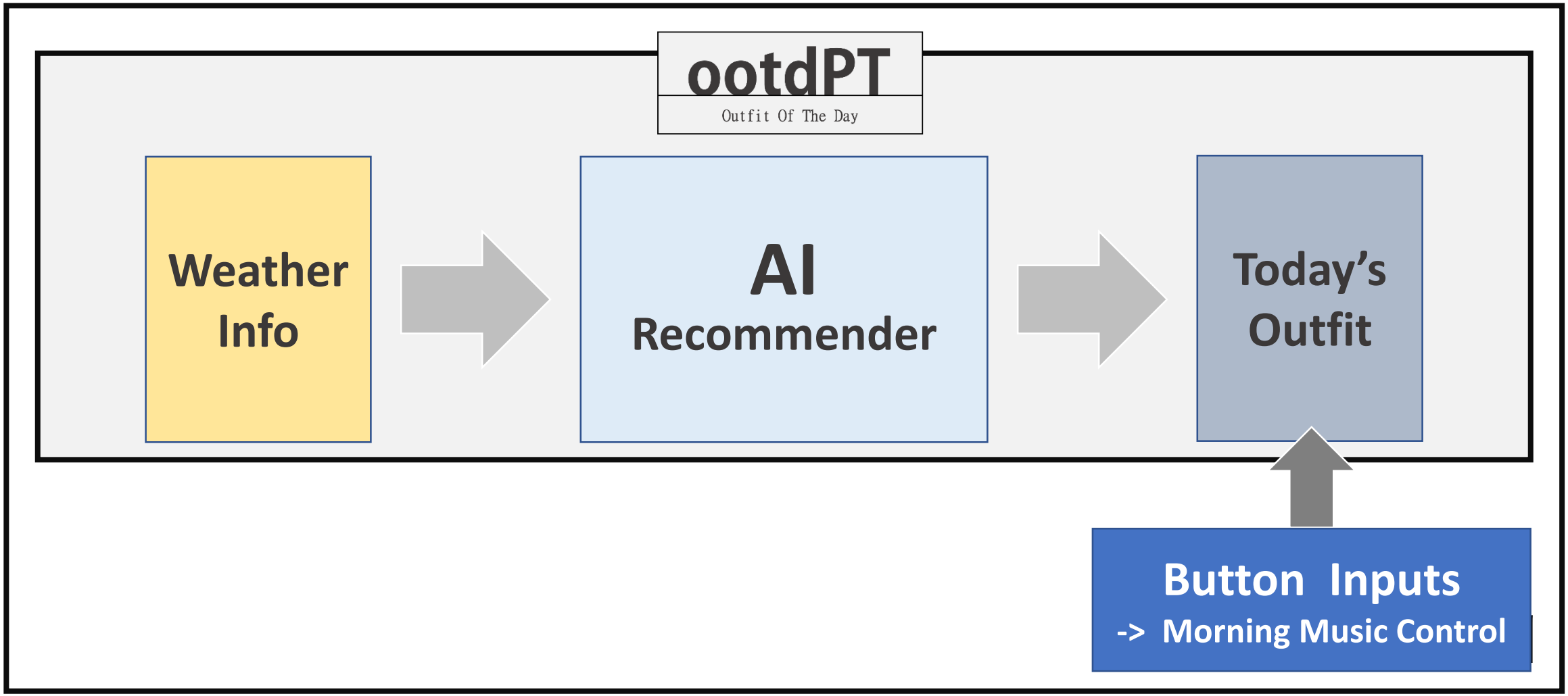
Idea


ootdPT
Outfit Of The Day
Weather Info
AI Recommender
Today's Outfit
ChatGPT
Achro-i.mx6q

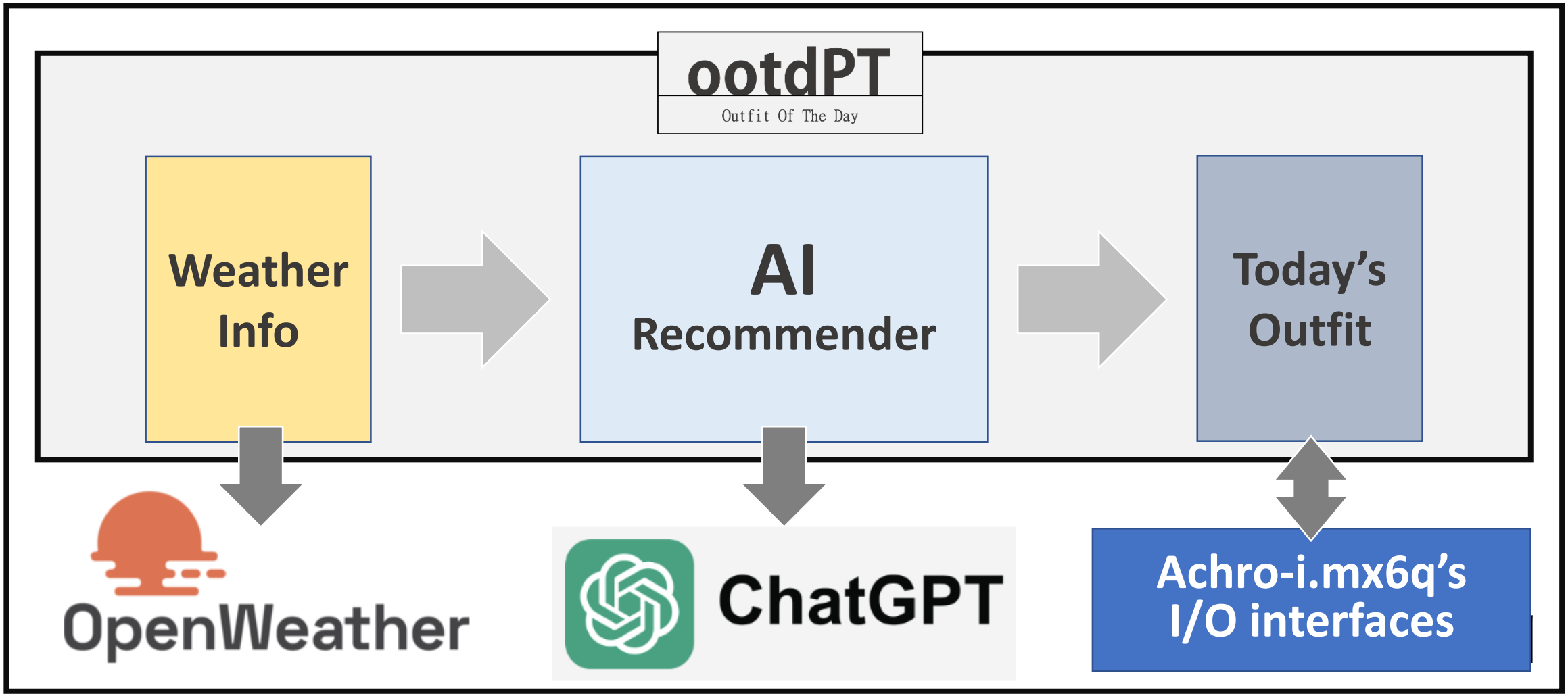
Idea


ootdPT
Outfit Of The Day
Weather Info
AI Recommender
Today's Outfit
FND, TEXT_LCD, LED, DOT and Device Screen

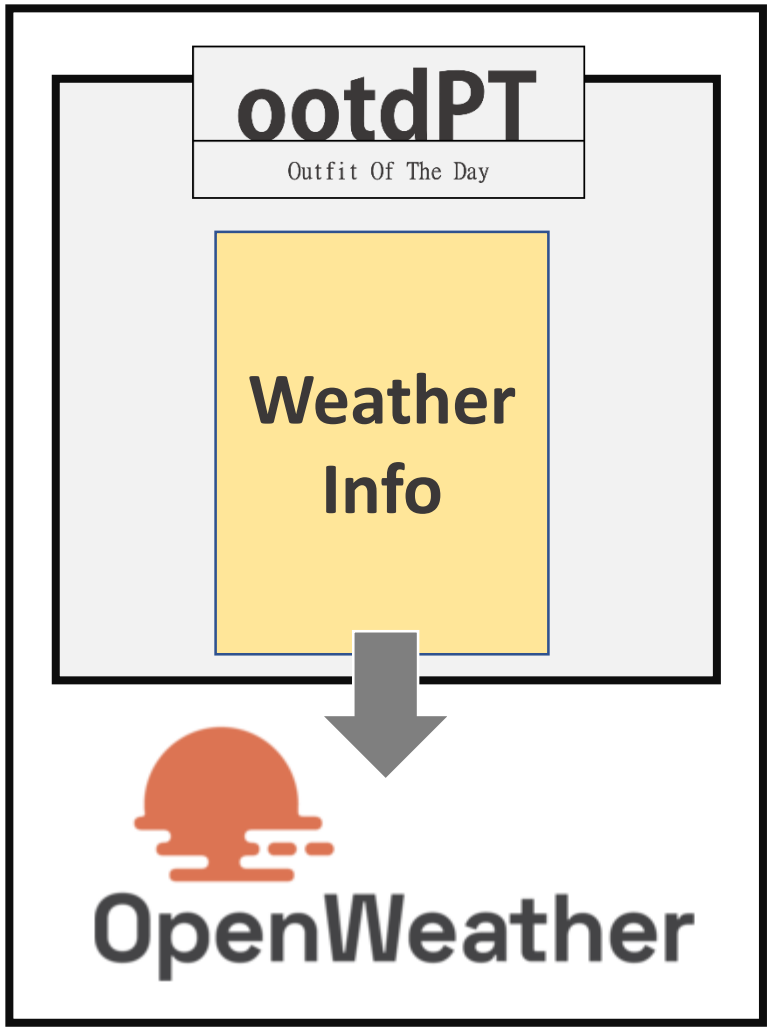
Idea


ootdPT
Outfit Of The Day
Weather Info
AI Recommender
Today's Outfit
Button Inputs
-> Morning Music Control

Overall Structure



Weather Information



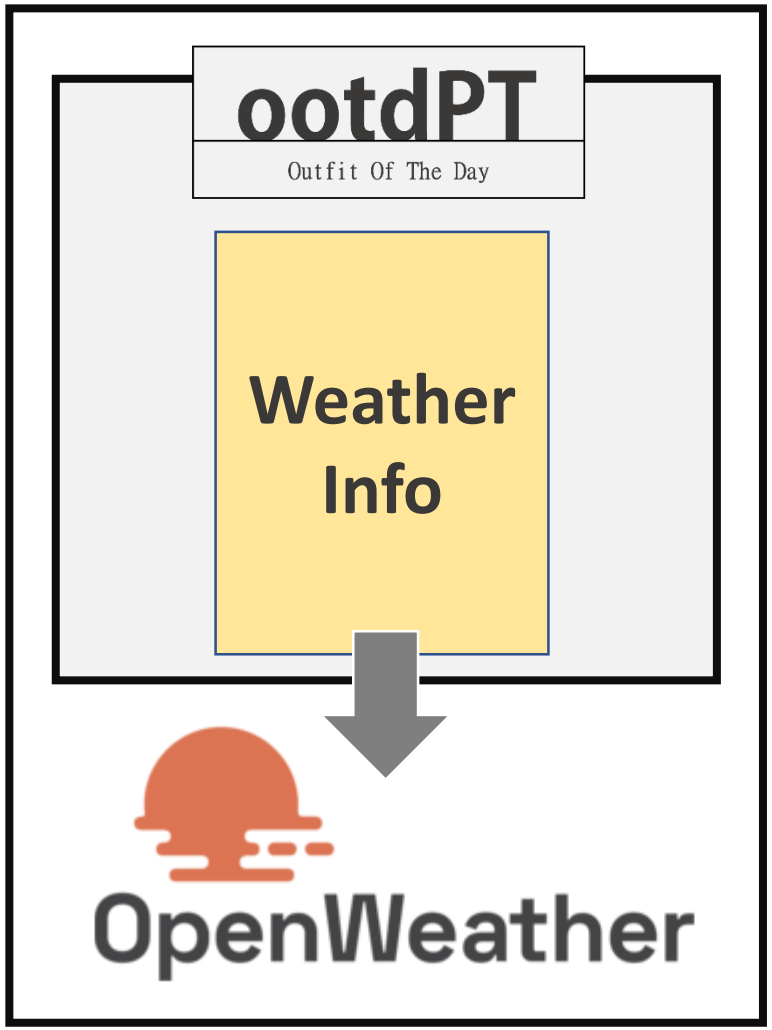
MainActivity

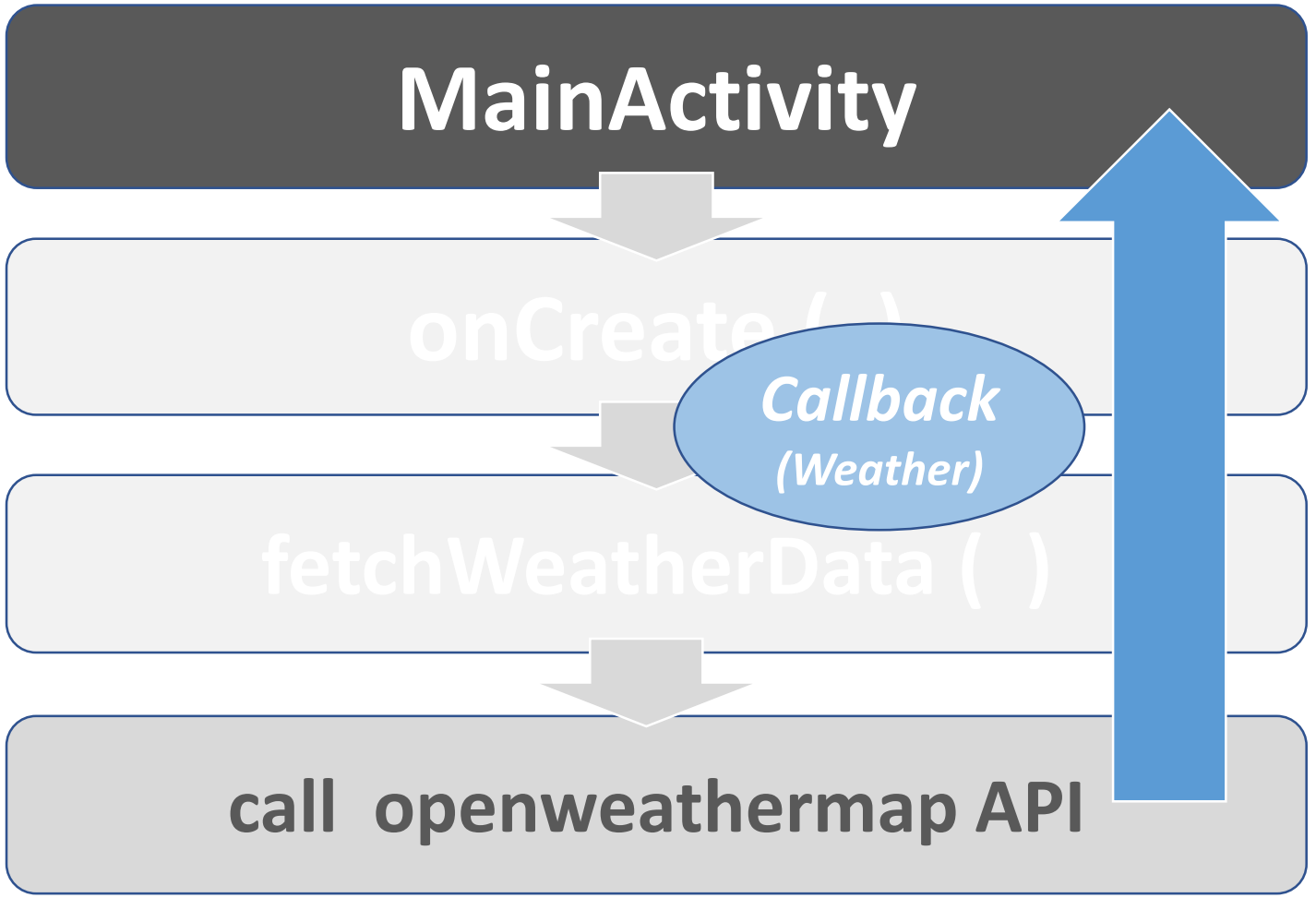
onCreate ( )

fetchWeatherData ( )

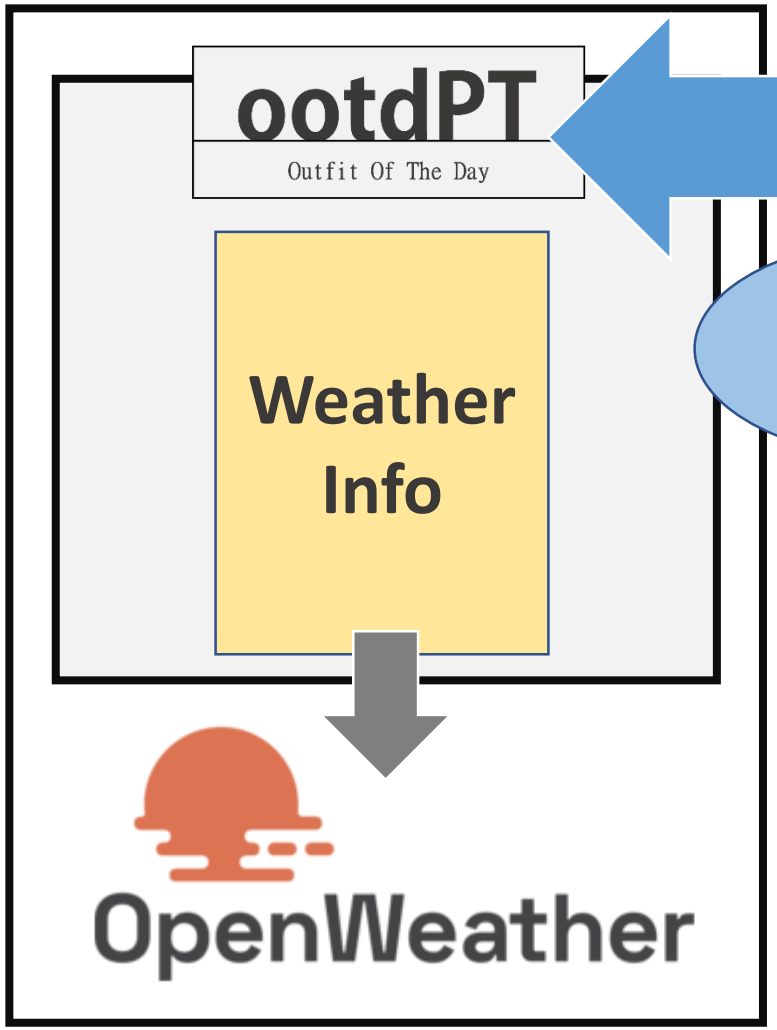
call openweathermap API

Weather Information





Weather Information



Pass to chatGPT

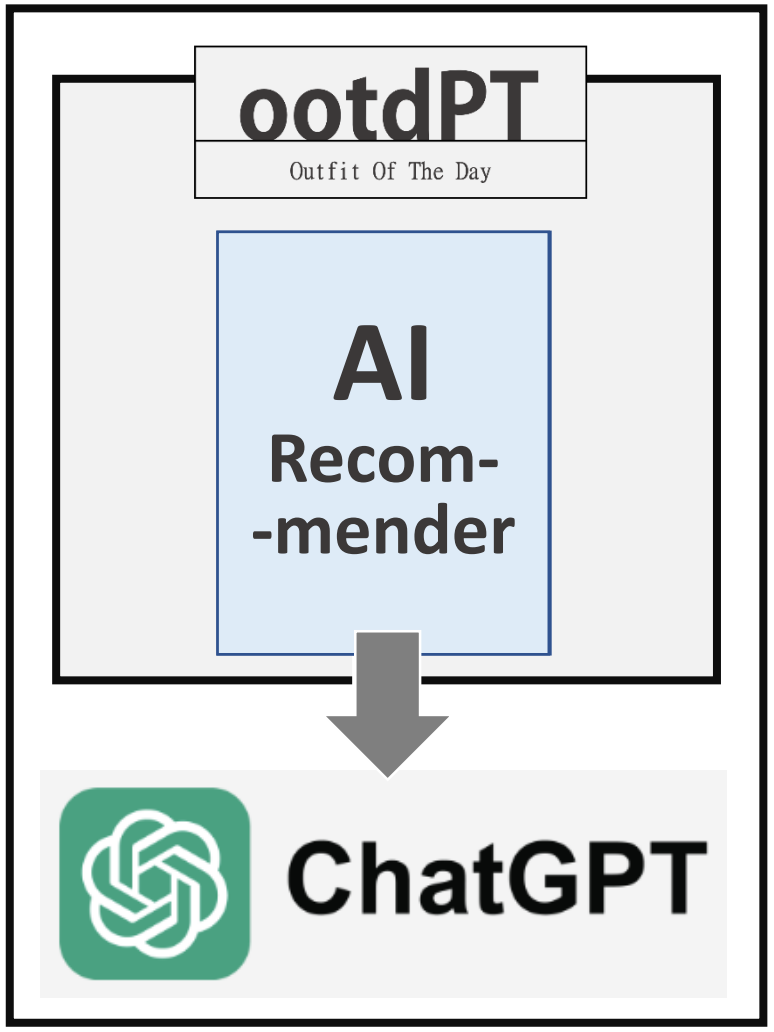
MainActivity

onCreate ( )

fetchWeatherData ( )

call openweathermap API

Outfit Recommendation

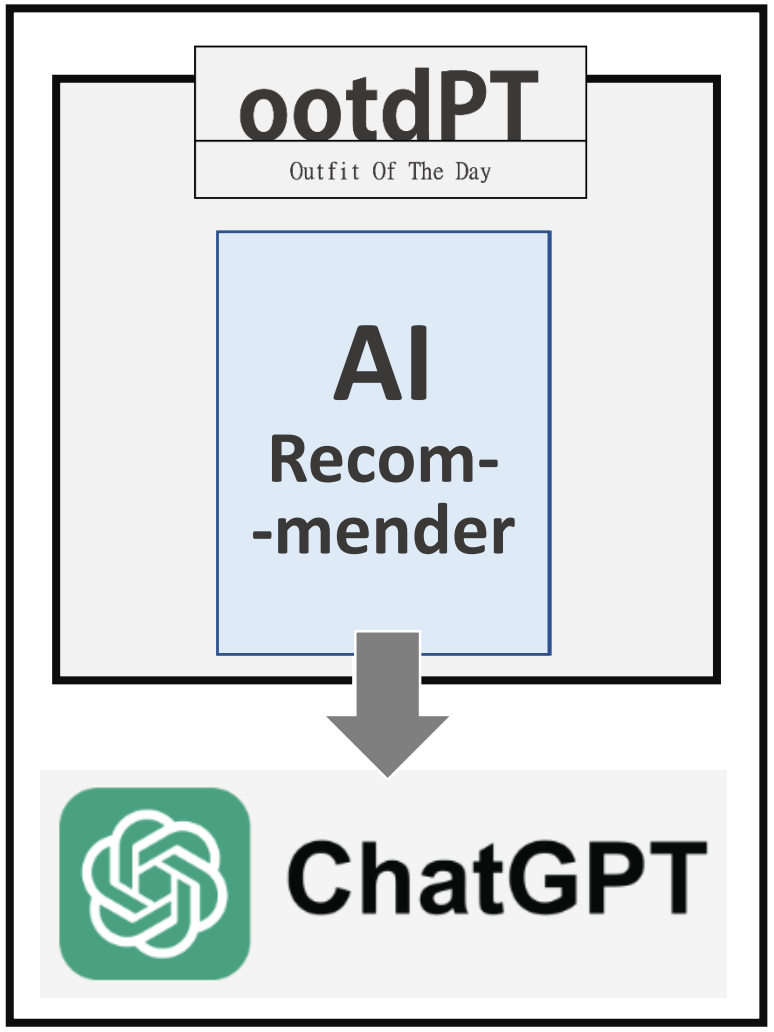


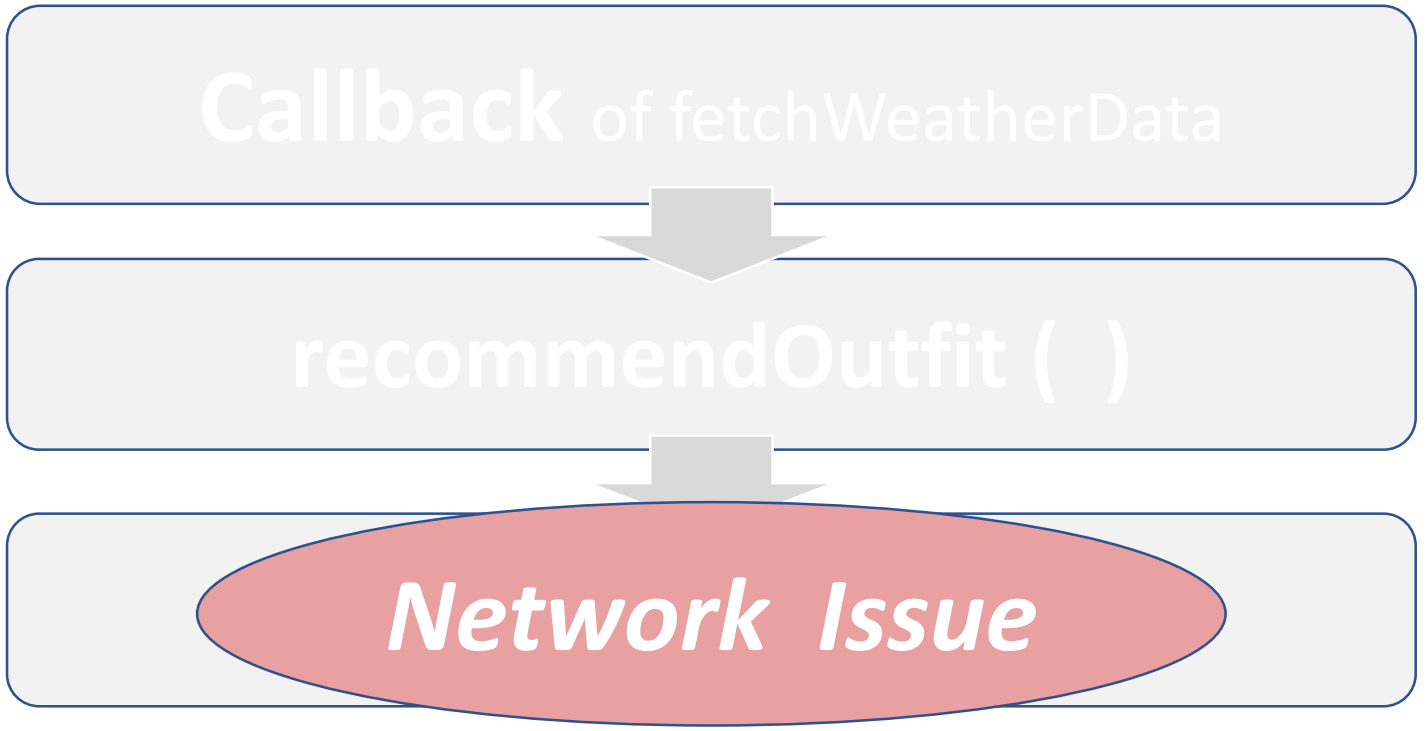
Callback of fetchWeatherData

recommendOutfit ( )

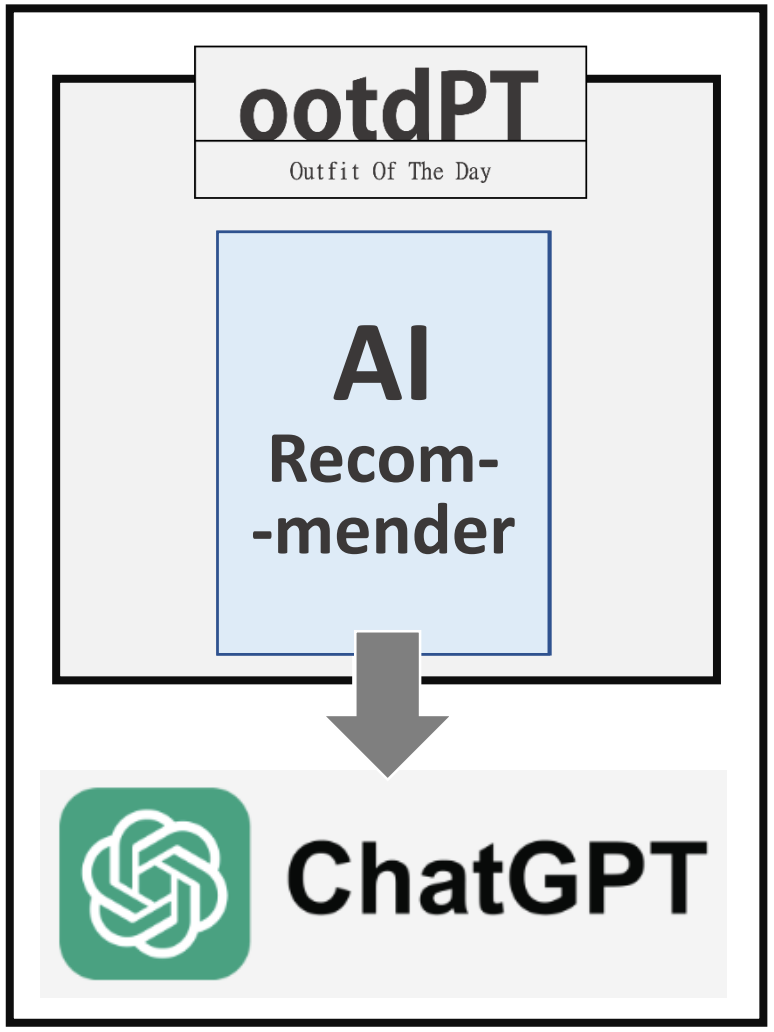
call chatGPT API

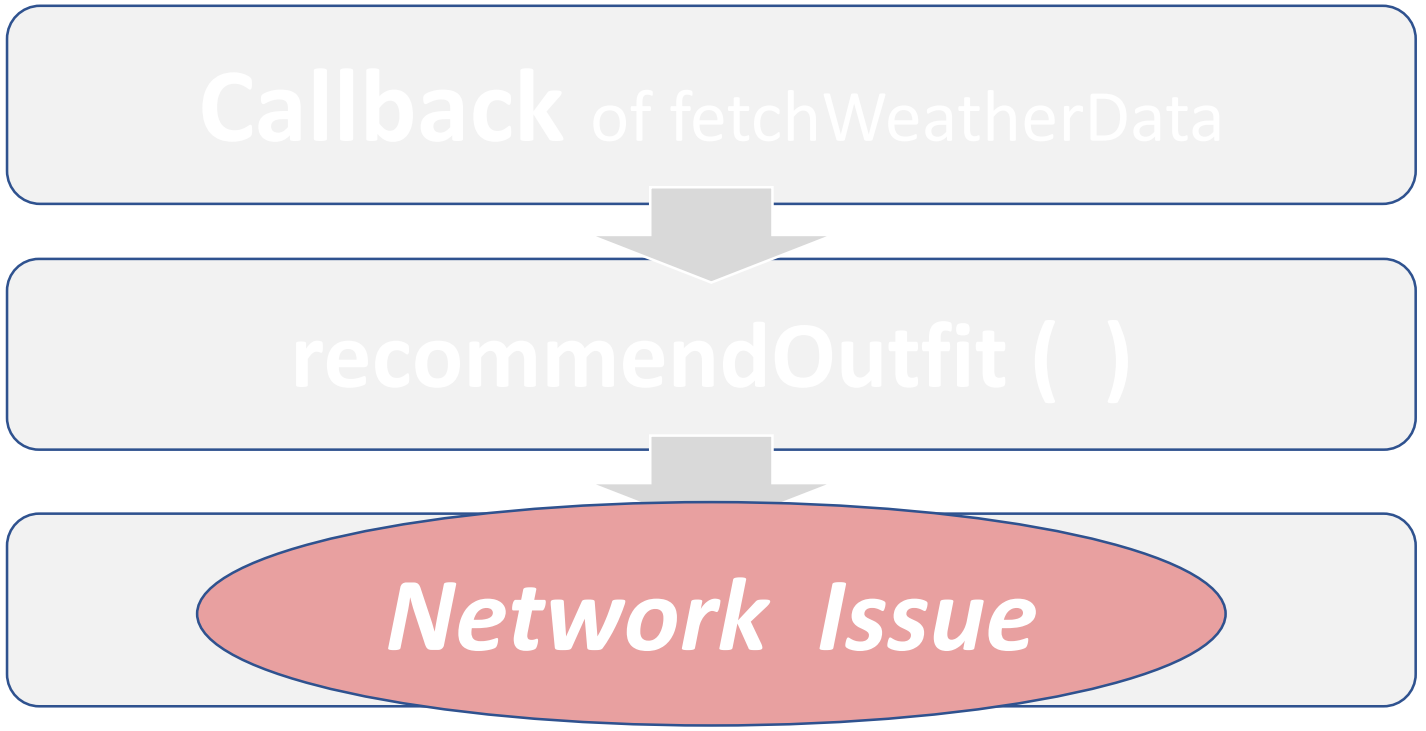
Outfit Recommendation





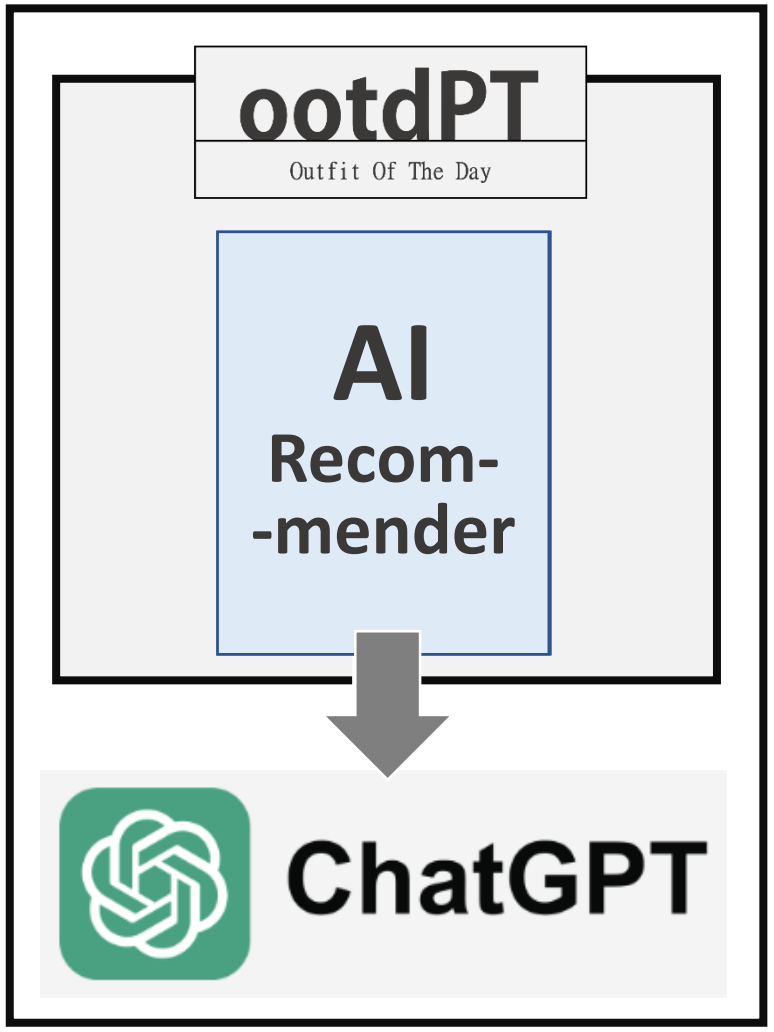
Outfit Recommendation





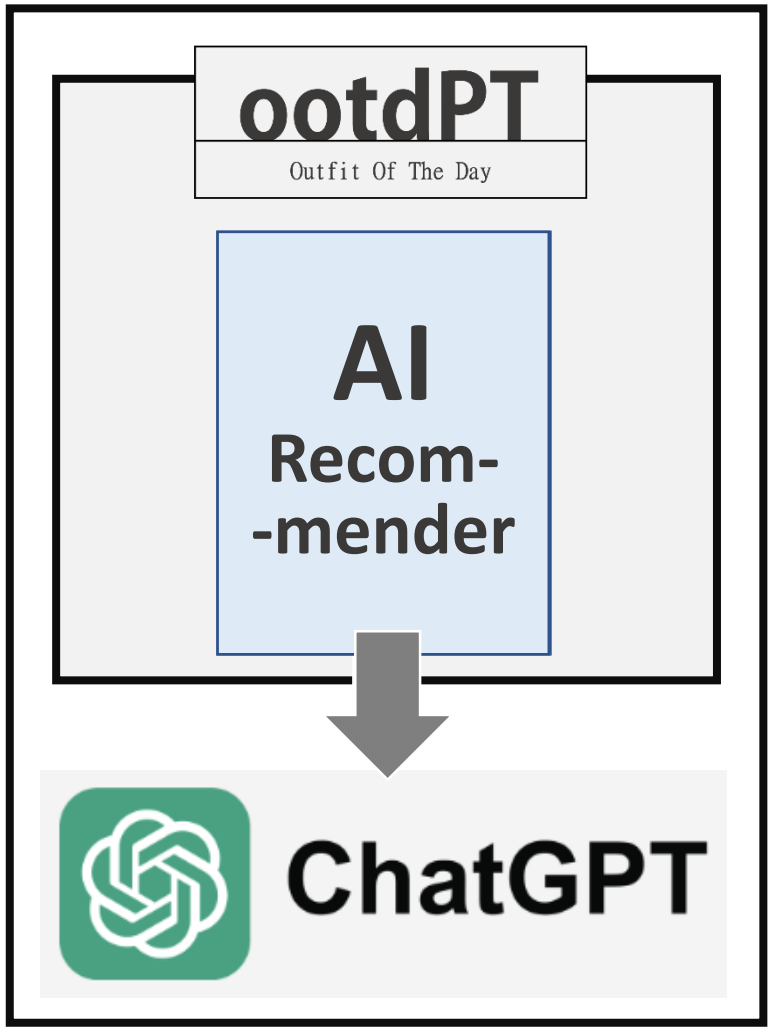
Android version4 **doesn't support** a network connection to chatGPT API

Outfit Recommendation



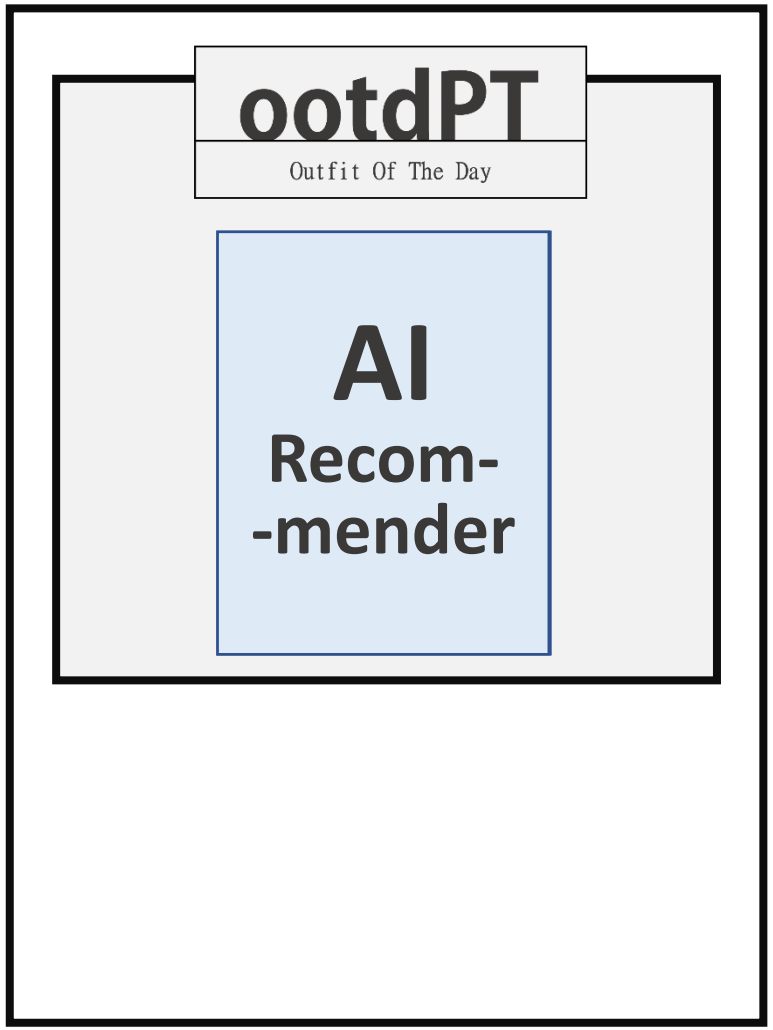
- **Option1)** In-device AI-like Recommendation Engine
- **Option2)** Intermediate Server Approach (Serial Port)
- **Option3)** Device Android Version Upgrade

Outfit Recommendation



- **Option1)** In-device chatGPT-like Recommendation Engine
- **Option2)** Intermediate Server Approach (Serial Port)
- **Option3)** Device Android Version Upgrade

Outfit Recommendation – Option1



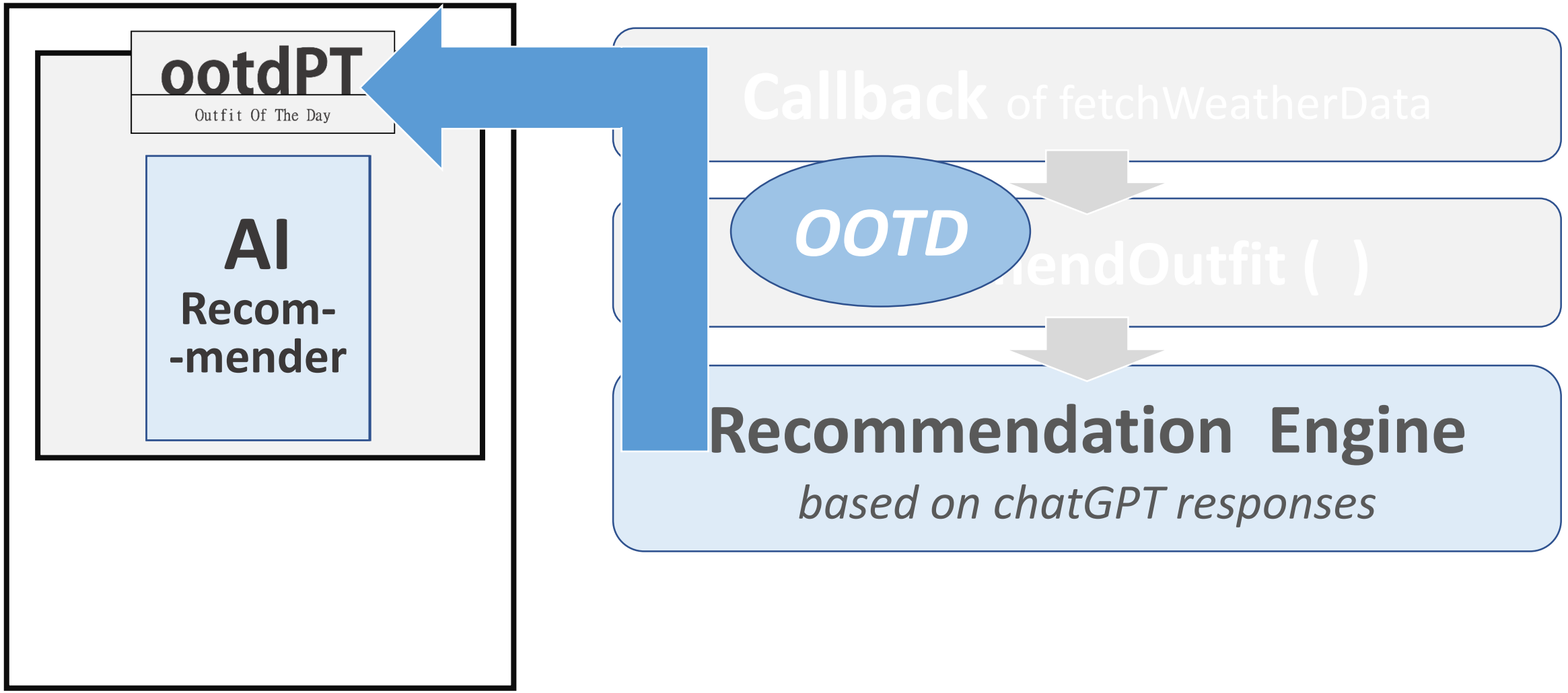
Callback of fetchWeatherData

recommendOutfit ( )

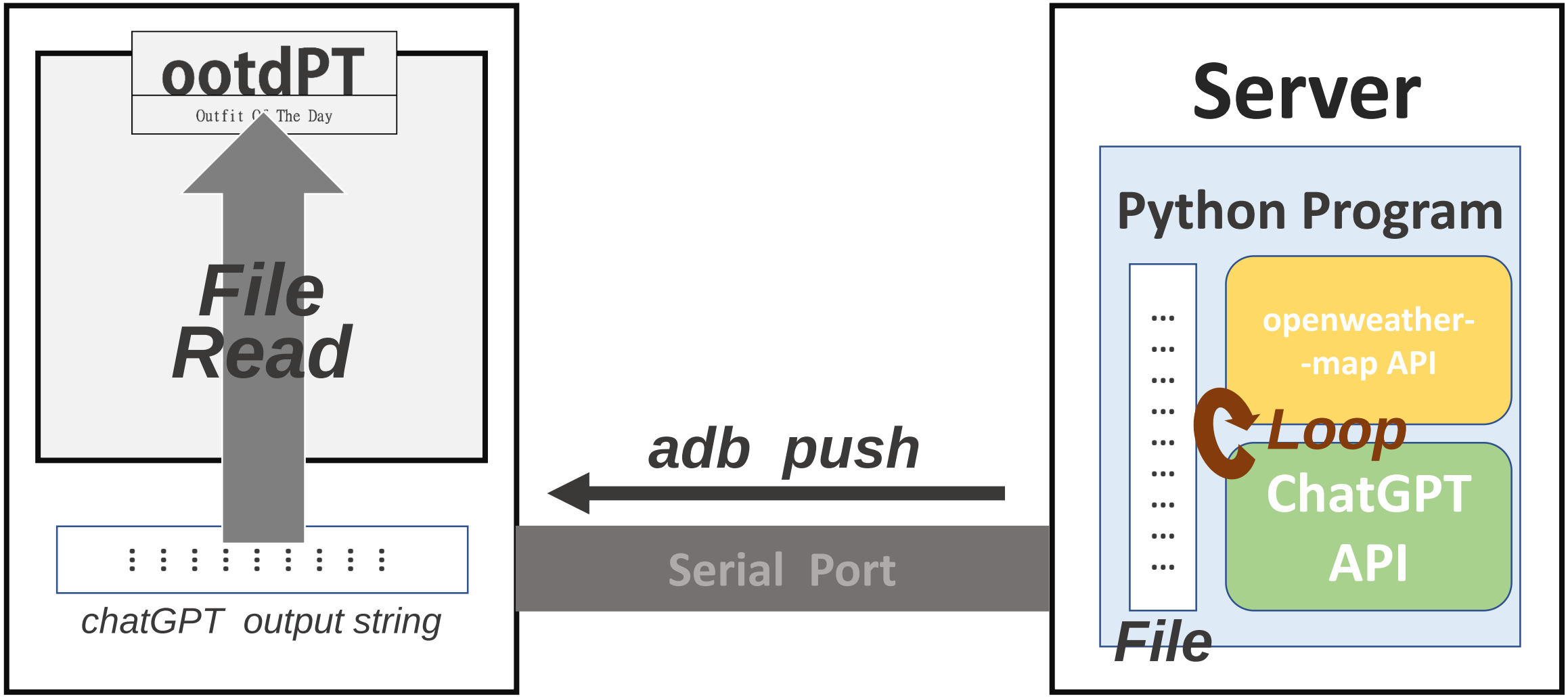
Recommendation Engine

based on chatGPT responses

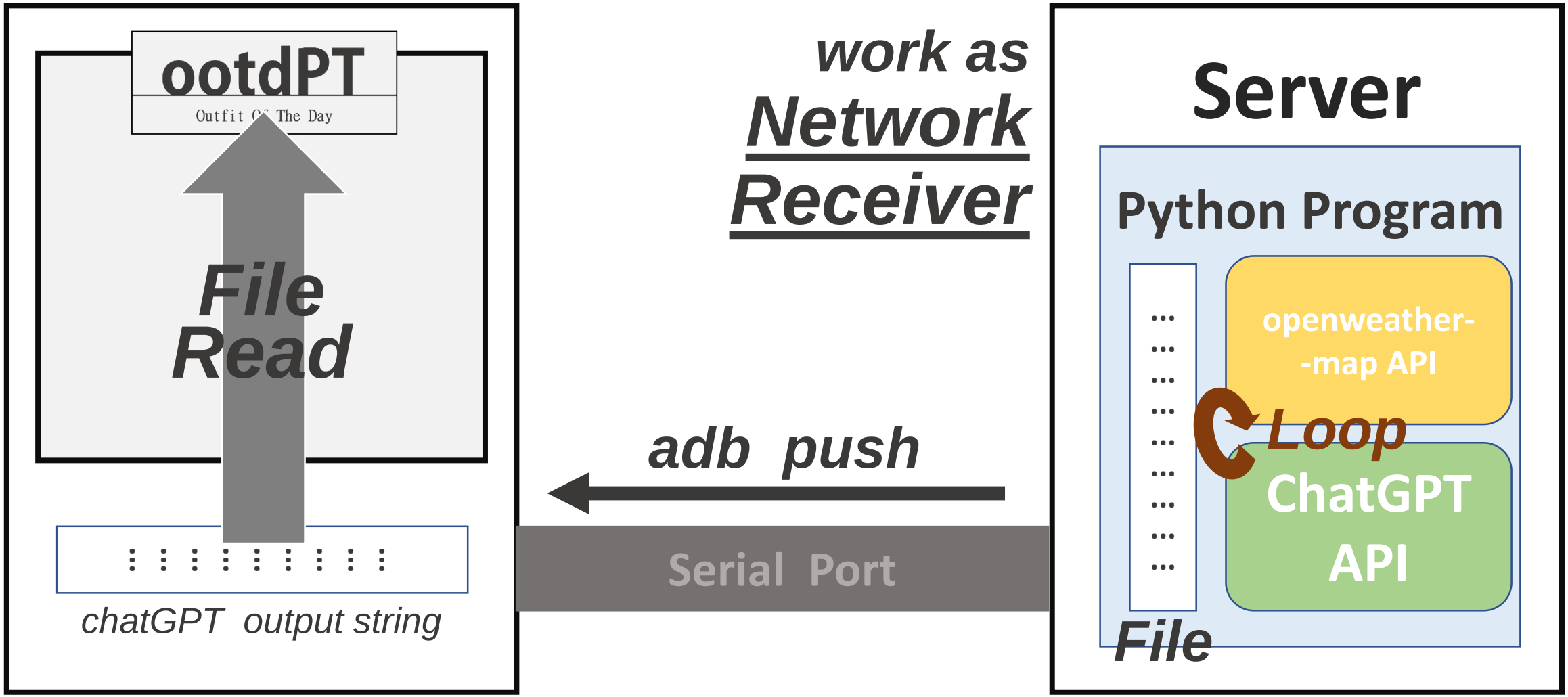
Outfit Recommendation – Option1



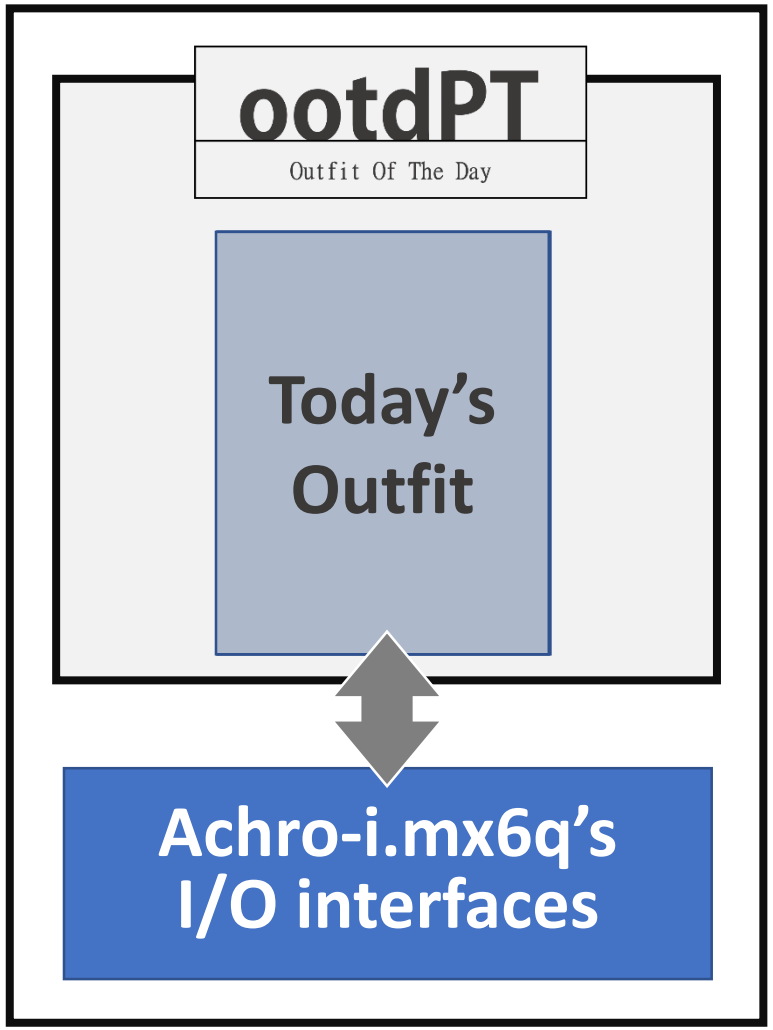
Outfit Recommendation – Option2



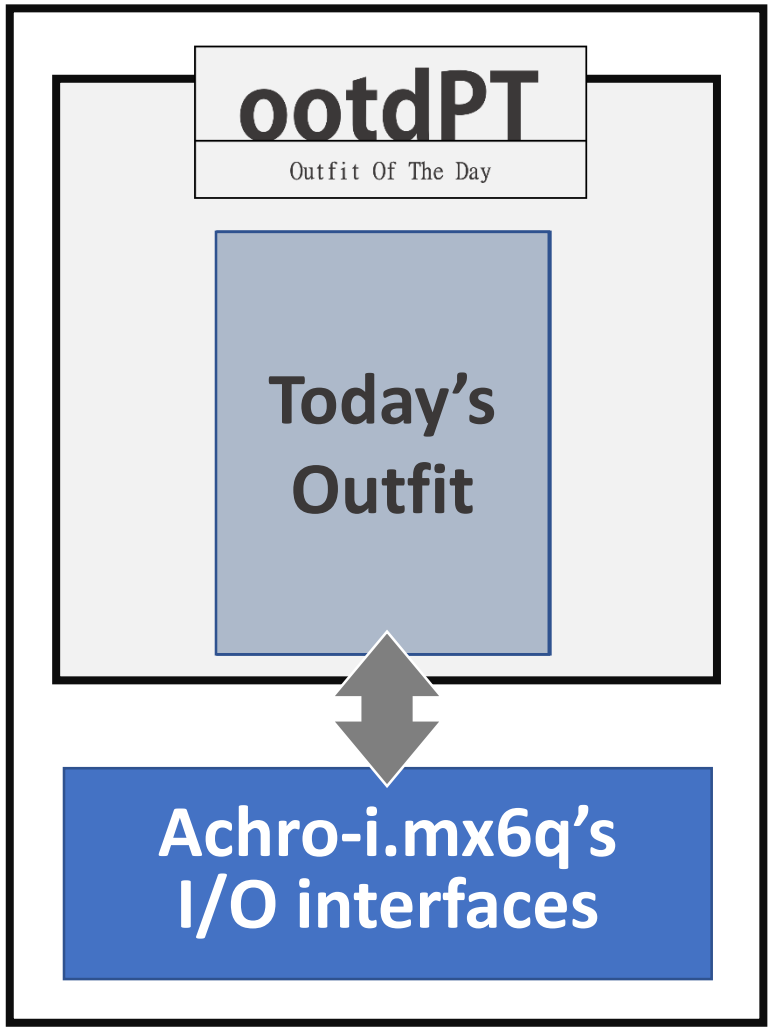
Outfit Recommendation – Option2



I/O with Device



I/O with Device

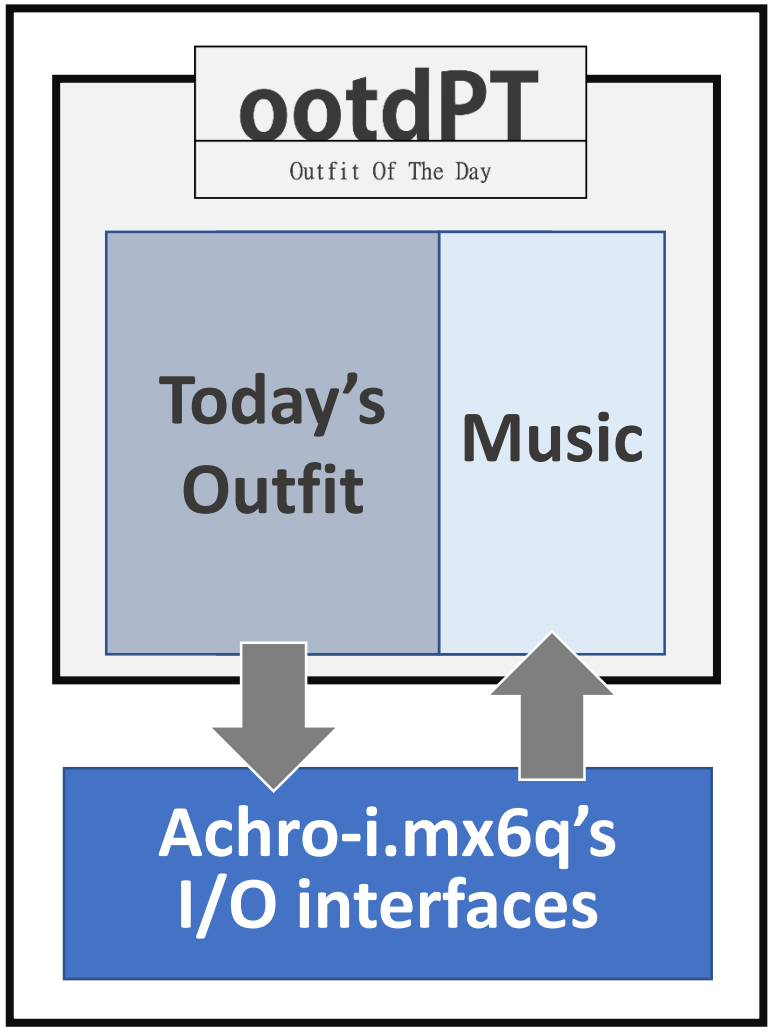


Additional Feature

Background Music

(since it targets 'home intercom')

I/O with Device

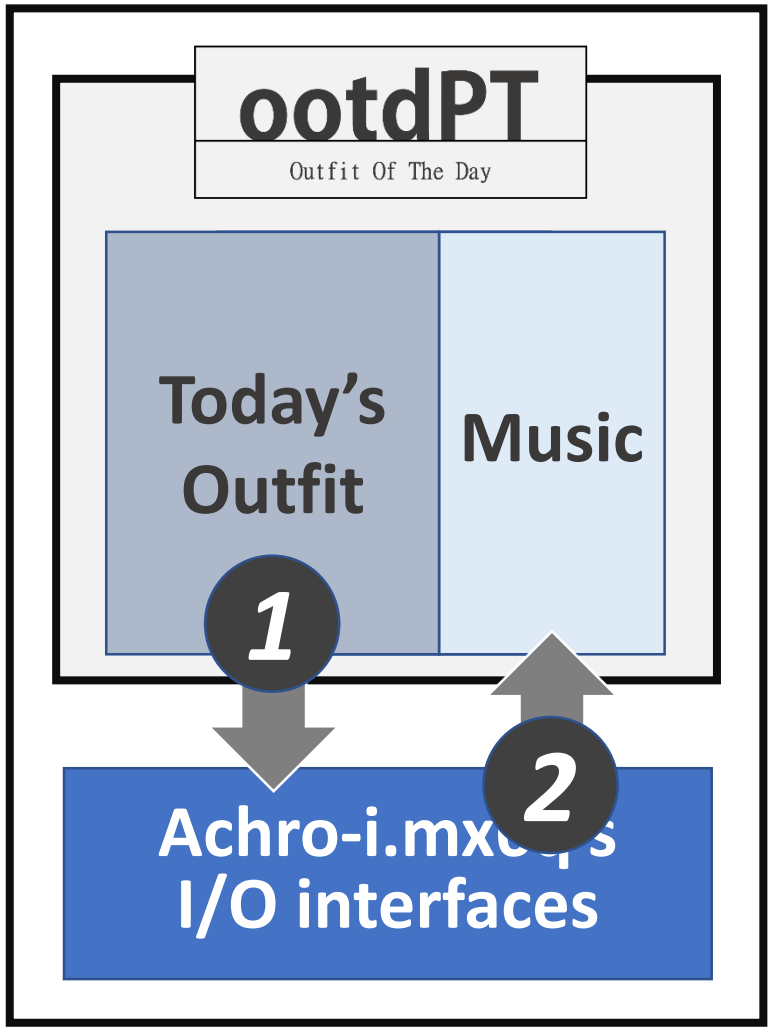


Additional Feature

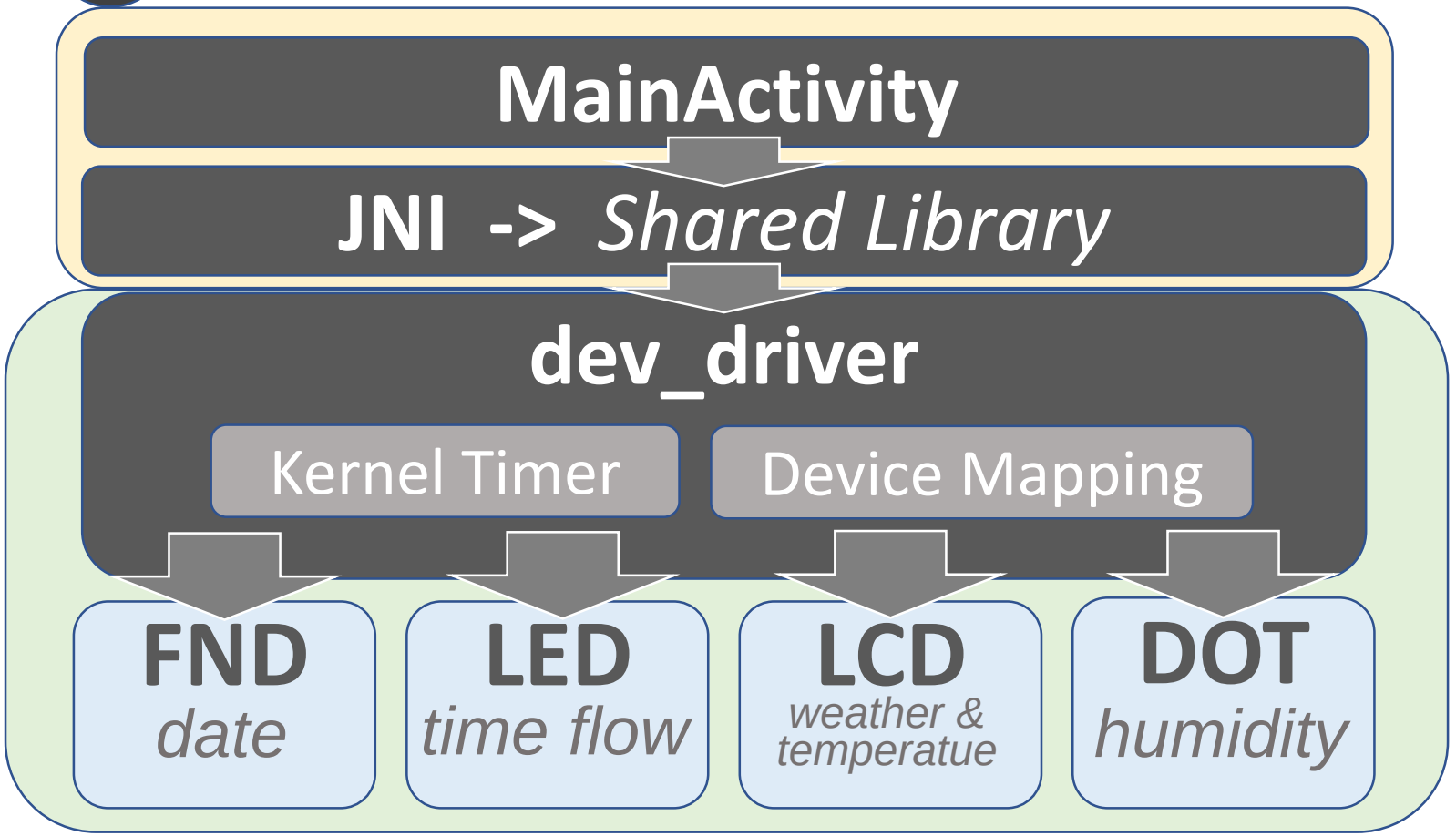
Background Music

(since it targets 'home intercom')

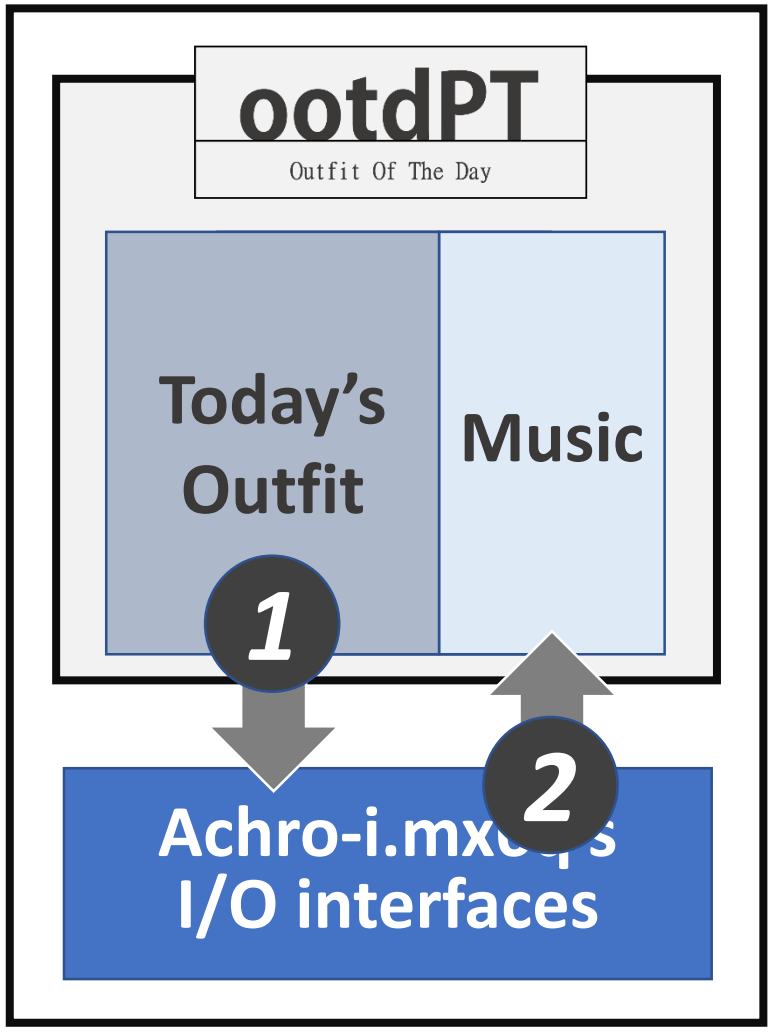
I/O with Device



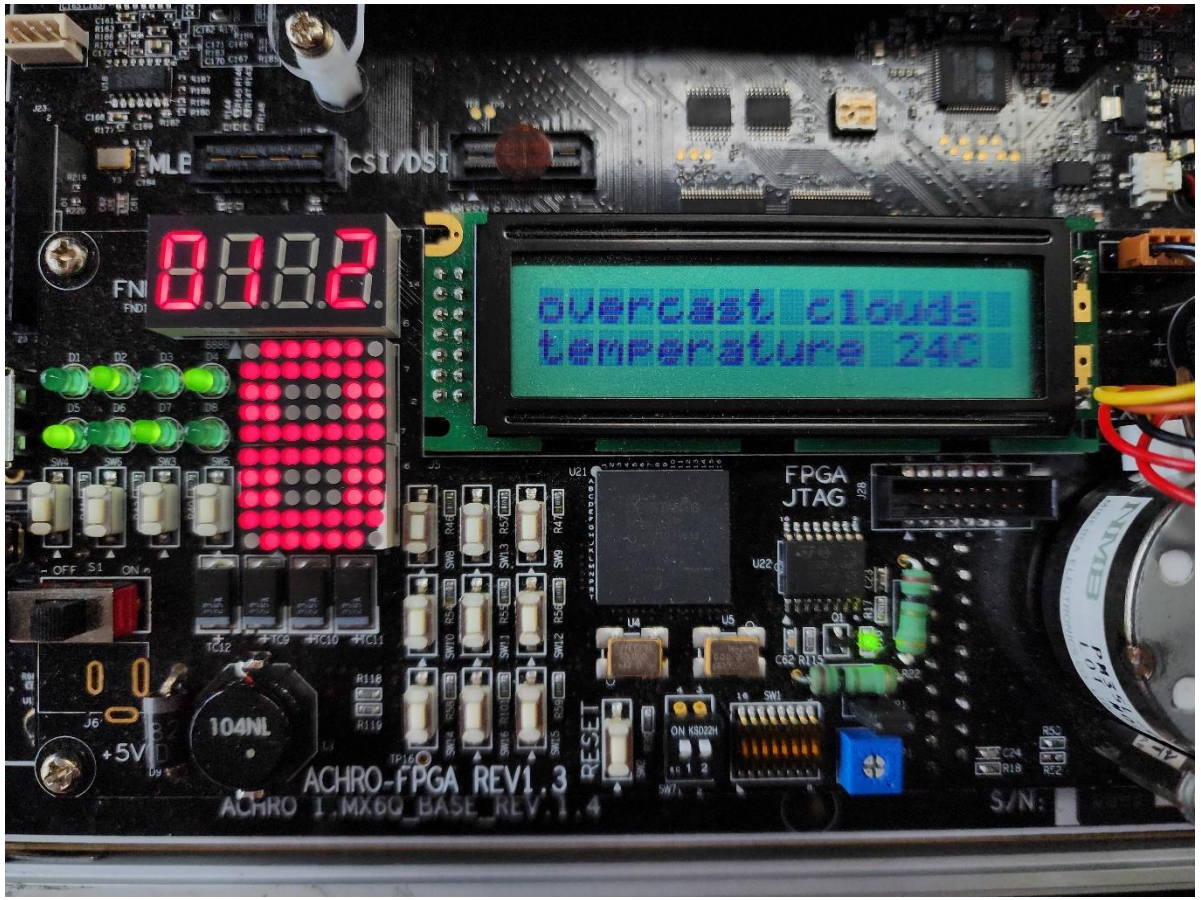
1 *Print weather information*



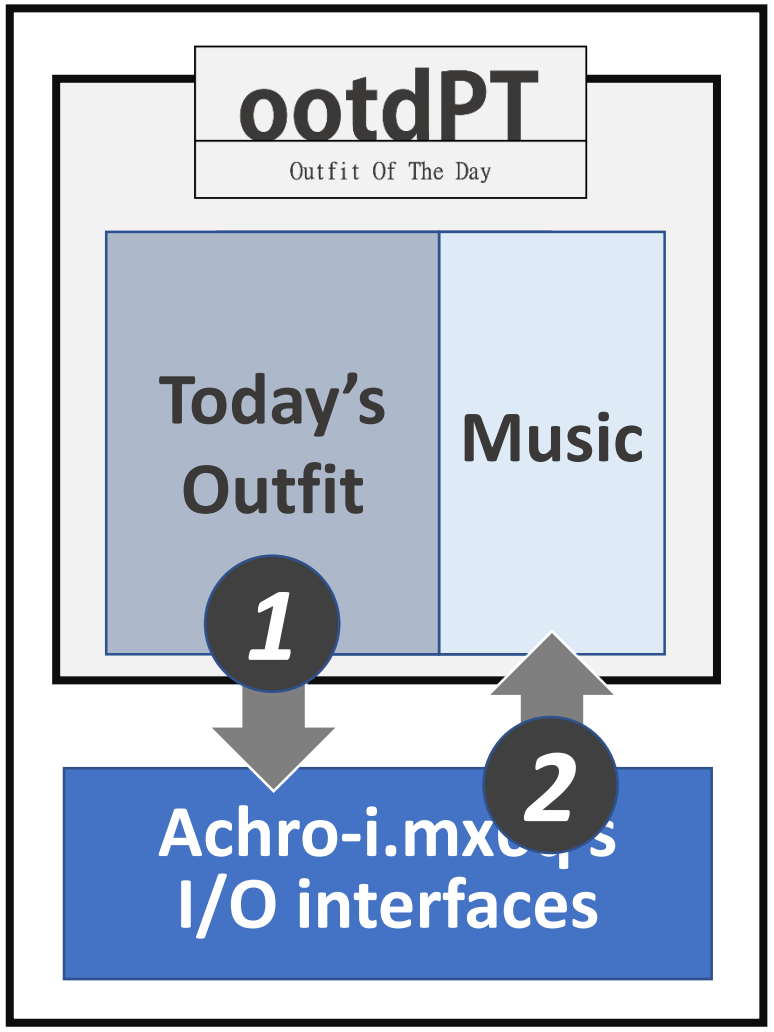
I/O with Device



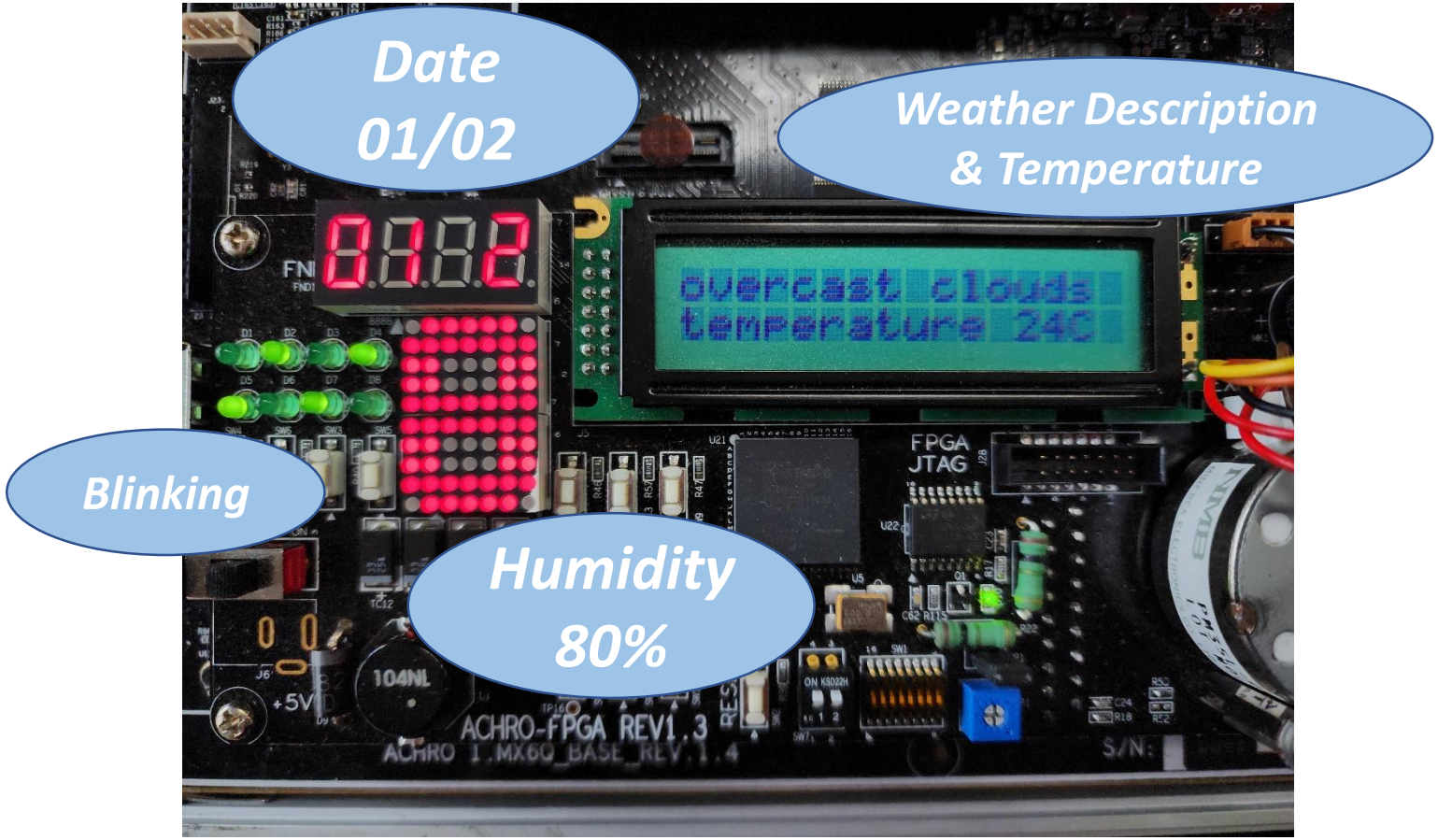
1 *Print weather information*



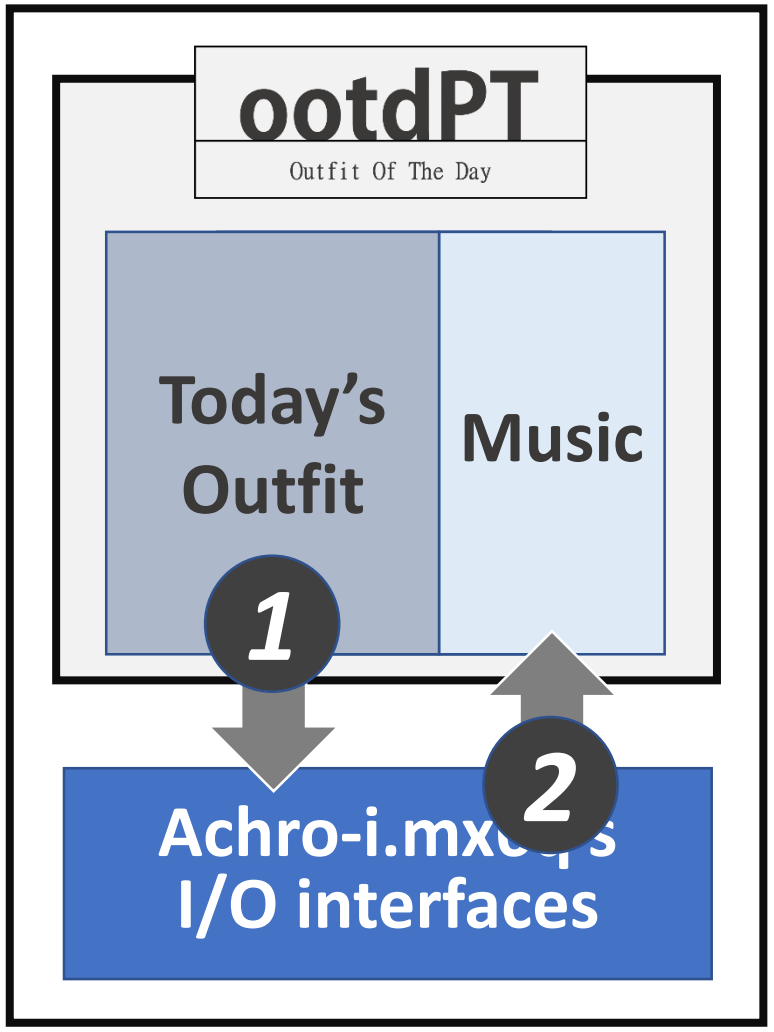
I/O with Device



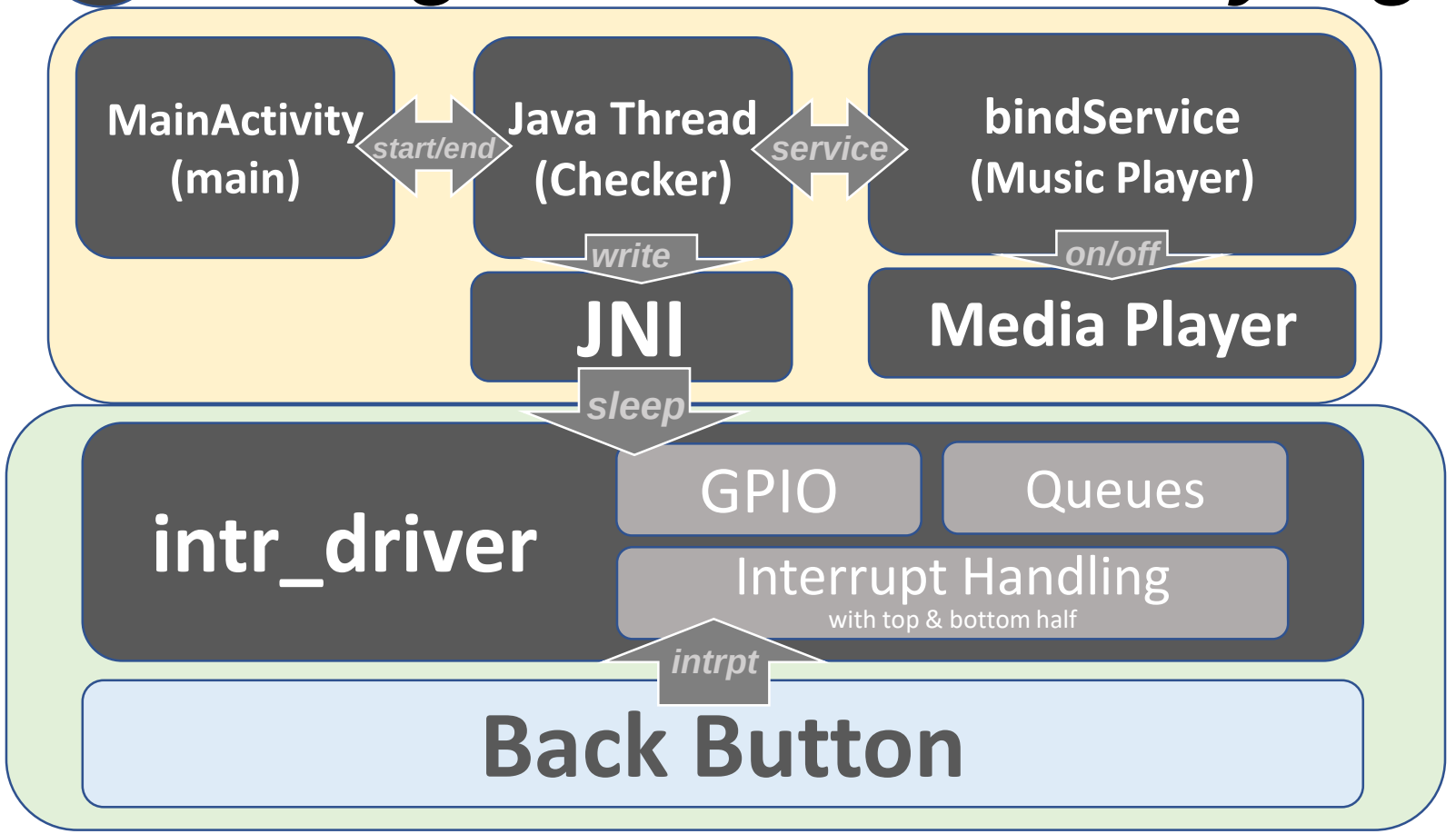
1 *Print weather information*



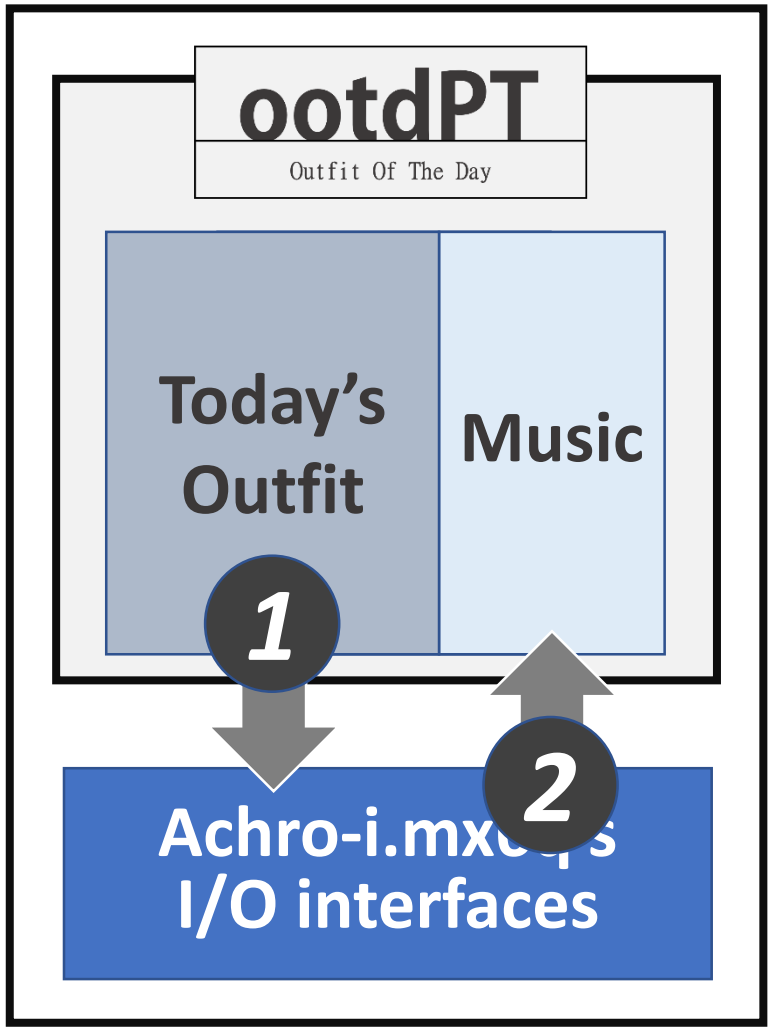
I/O with Device



2 Background Music Playing



I/O with Device

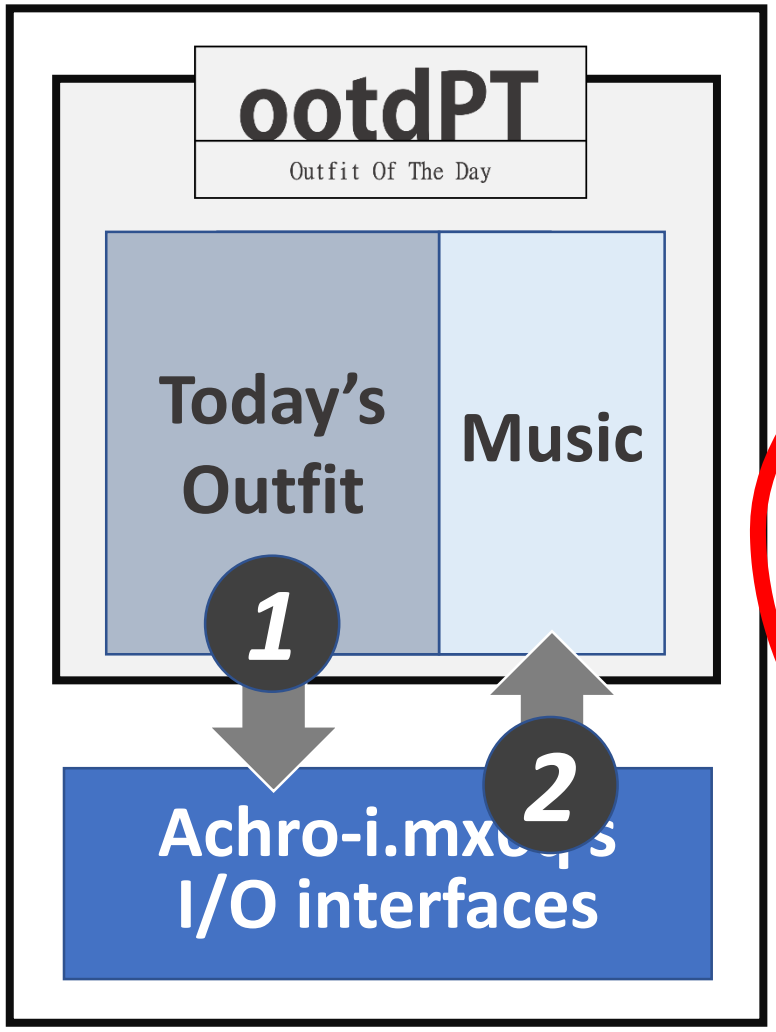


2 *Background Music Playing*

```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

I/O with Device

2 Background Music Playing



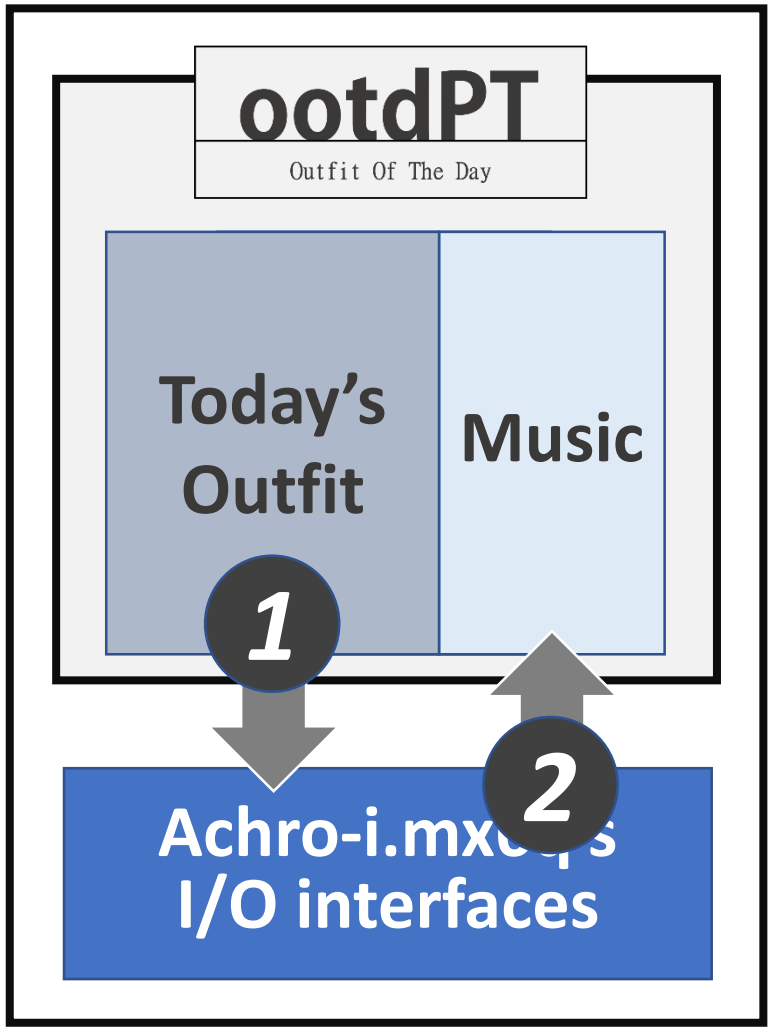

```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void
                                                    IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        viceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

```
public void run() {
    while (isRunning) {
            writeDevice(fd);  // It will sleep here!
        toggleMusic();    // After the thread woke up
    }
}
```

```
public void startMusic() {
    mp.start();
}

public void stopMusic() {
    mp.pause();
}
```

I/O with Device

2 Background Music Playing



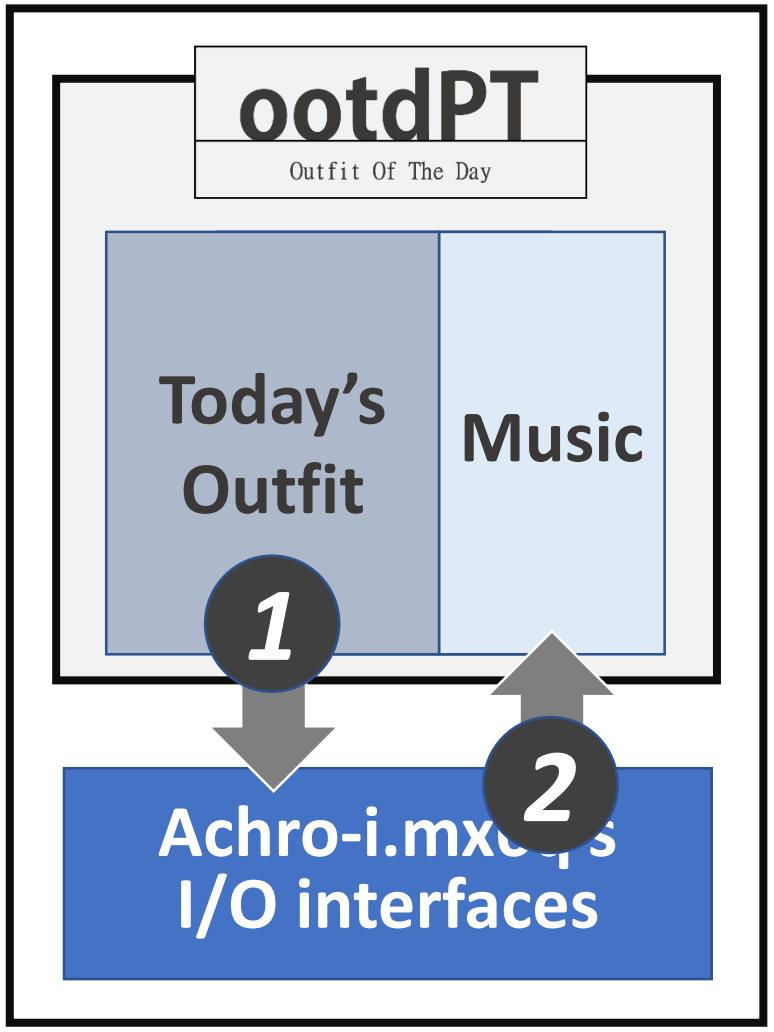

```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the t  n on/off thread after the service is bound
        turnOnOff hread = new TurnOnOffThre d(myService);
        turnO  ffThread.start();
    }
```

```
public void run() {
    whil  isRunning) {
        writeDevice(fd);    // It will sleep here!
        toggleMusic();      // After the thread woke up
    }
}
```

```
public void startMusic() {
    mp.start();
}

public void stopMusic() {
    mp.pause();
}
```

```
/* write() Handler for this driver */
static int intr_driver_write(
    struct file *file, const char __user *buf, size_t count, loff_t *f_pos
) {
    printk("[INTR_DRIVER] %s called\n", __func__);

    /* Put the user thread on a sleep */
    printk("[INTR_DRIVER] Make user thread(java thread) sleep\n");
    interruptible_sleep_on(&intrq);

    return 0;
}
```

I/O with Device



2 Background Music Playing

```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

```
public void run() {
    while (isRunning) {
            writeDevice(fd);  // It will sleep here!
        toggleMusic();    // After the thread woke up
    }
}
```

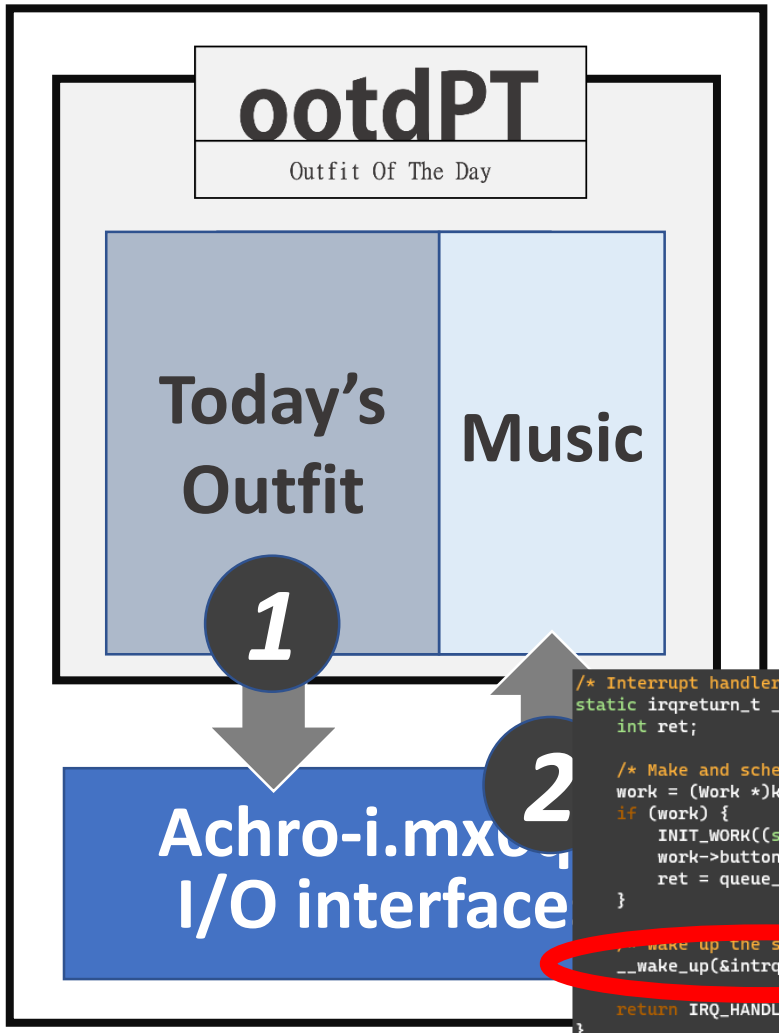
```
public void startMusic() {
    mp.start();
}

public void stopMusic() {
    mp.pause();
}
```

```
/* write() Handler for this driver */
static int intr_driver_write(
    struct file *file, const char __user *buf, size_t count, loff_t *f_pos
) {
    printk("[INTR_DRIVER] %s called\n", __func__);

    /* Put the user thread on a sleep */
    thread(java thread) sleep\n");
    interruptible_sleep_on(&intrq);

    return 0;
}
```

Sleep

I/O with Device

2 Background Music Playing




```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

```
public void run() {
    while (isRunning) {
            writeDevice(fd);  // It will sleep here!
        toggleMusic();    // After the thread woke up
    }
}
```

```
/* Interrupt handler for 'back' button, it will trigger the sleeping java thread */
static irqreturn_t _back_button_handler(int irq, void* dev_id) {
    int ret;

    /* Make and schedule the bottom half (printks) */
    work = (Work *)kmalloc(sizeof(Work), GFP_KERNEL);
    if (work) {
        INIT_WORK((struct work_struct *)work, work_function);
        work->button = BACK;
        ret = queue_work(workq, (struct work_struct *)work);
    }

    /* Wake up the sleeping thread (which is a java thread in here) */
    __wake_up(&intrq, 1, 1, NULL);

    return IRQ_HANDLED;
}
```

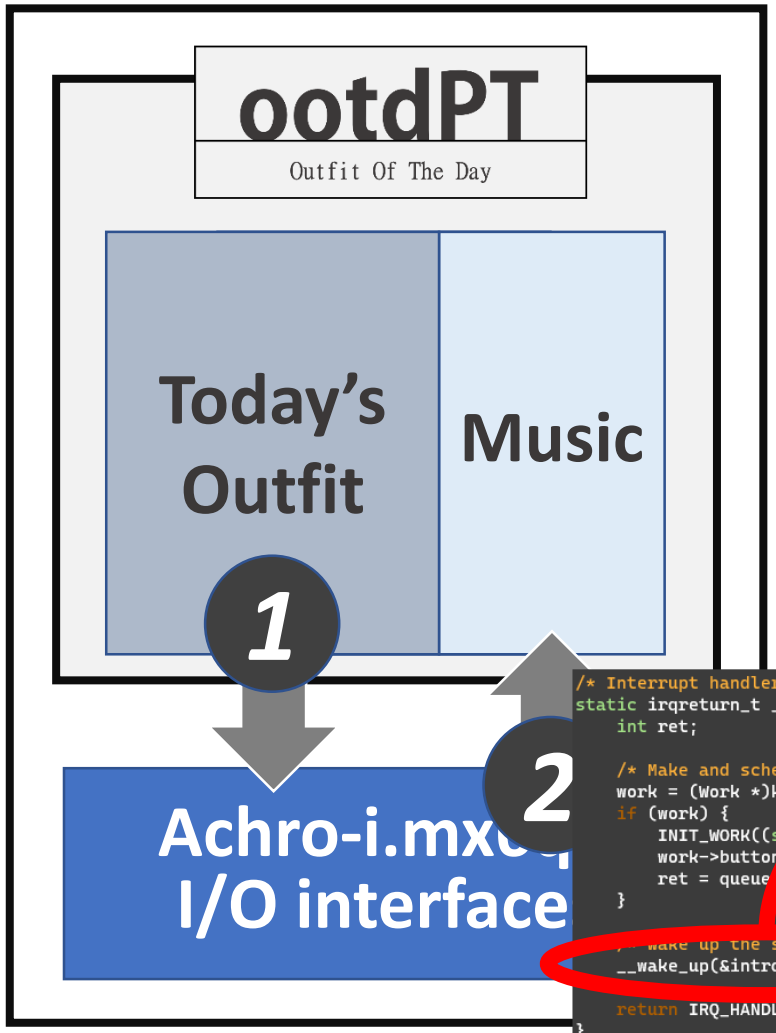
Wake up

if 'back' clicked and interrupt generated

```
/* write() Handler for this driver */
static int intr_driver_write(
    struct file *file, const char __user *buf, size_t count, loff_t *f_pos
) {
    printk("[INTR_DRIVER] %s called\n", __func__);

    /* Put the user thread on a sleep */
    printk("[INTR_DRIVER] Make user thread(java thread) sleep\n");
    interruptible_sleep_on(&intrq);

    return 0;
}
```

I/O with Device

2 Background Music Playing




```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

```
public void run() {
    while (isRunning) {
                                // It will sleep here!
        toggleMusic();          // thread woke up
    }
}
```

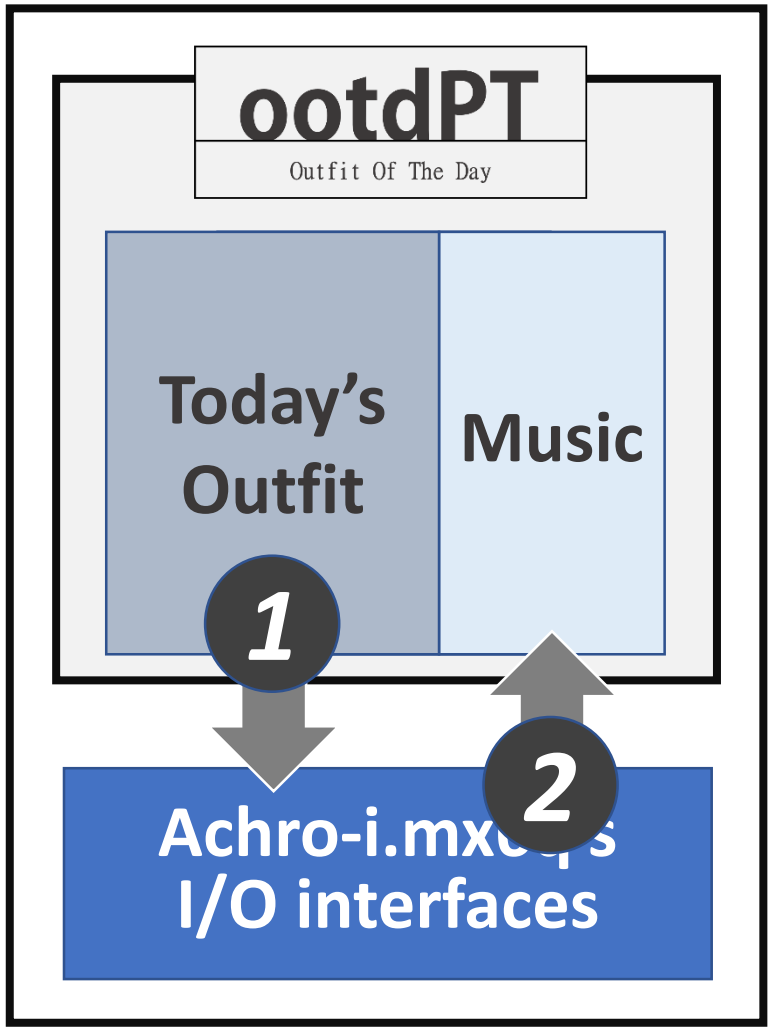
```
private void toggleMusic() {
    isMusicOn = !isMusicOn;
    // Control the music playback
    if (isMusicOn) {
        myService.startMusic();
        Log.d("TurnOnOffThread", "Music turned on");
    } else {
        myService.stopMusic();
        Log.d("TurnOnOffThread", "Music turned off");
    }
}
```

```
/* Interrupt handler for 'back' button, it will trigger the sleeping java thread */
static irqreturn_t _back_button_handler(int irq, void* dev_id) {
    int ret;

    /* Make and schedule the bottom half (printks) */
    work = (Work *)kmalloc(sizeof(Work), GFP_KERNEL);
    if (work) {
        INIT_WORK((struct work_struct *)work, work_function);
        work->button = BACK;
        ret = queue_work(workq, (struct work_struct *)work);
    }

    /* Wake up the sleeping thread (which is a java thread in here) */
    __wake_up(&intrq, 1, 1, NULL);

    return IRQ_HANDLED;
}
```

```
/* write() Handler for this driver */
static int intr_driver_write(
    struct file *file, const char __user *buf, size_t count, loff_t *f_pos
) {
    printk("[INTR_DRIVER] %s called\n", __func__);

    /* Put the user thread on a sleep */
    printk("[INTR_DRIVER] Make user thread(java thread) sleep\n");
    interruptible_sleep_on(&intrq);

    return 0;
}
```

I/O with Device

2 Background Music Playing




```
private ServiceConnection serviceConnection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName name, IBinder service) {
        MyService.MyBinder binder = (MyService.MyBinder) service;
        myService = binder.getService();
        isServiceBound = true;

        // Start the turn on/off thread after the service is bound
        turnOnOffThread = new TurnOnOffThread(myService);
        turnOnOffThread.start();
    }
```

```
public void run() {
    while (isRunning) {
            writeDevice(fd);  // It will sleep here!
        toggleMusic();    // After the thread woke up
    }
}
```

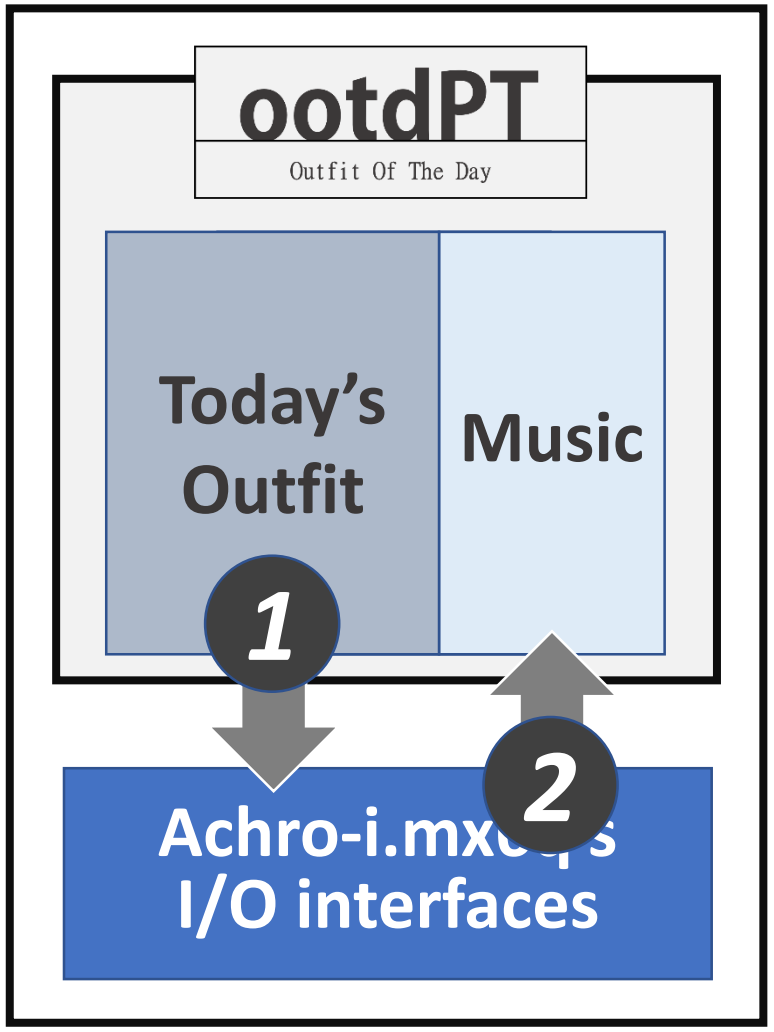
```
private void toggleMusic() {
    isMusicOn = !isMusicOn;
    // Control the music playback
    if (isMusicOn) {
        myService.startMusic();
        Log.d("TurnOnOffThread", "Music turned on");
    } else {
        myService.stopMusic();
        Log.d("TurnOnOffThread", "Music turned off");
    }
}
```

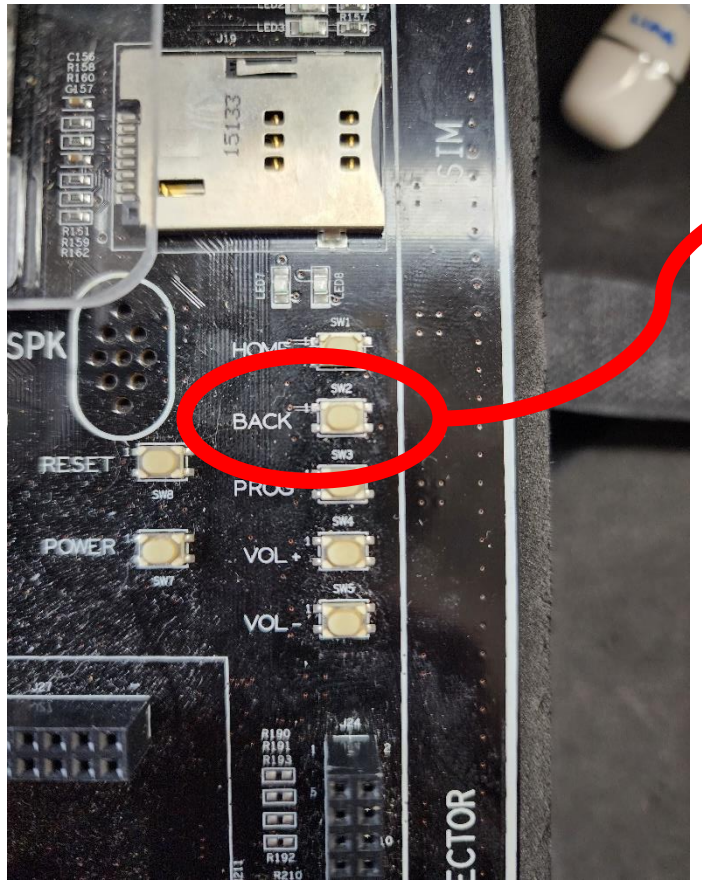
```
public void startMusic() {
    mp.start();
}

public void stopMusic() {
    mp.pause();
}
```

```
/* write() Handler for this driver */
static int intr_driver_write(
    struct file *file, const char __user *buf, size_t count, loff_t *f_pos
) {
    printk("[INTR_DRIVER] %s called\n", __func__);

    /* Put the user thread on a sleep */
    printk("[INTR_DRIVER] Make user thread(java thread) sleep\n");
    interruptible_sleep_on(&intrq);

    return 0;
}
```

I/O with Device

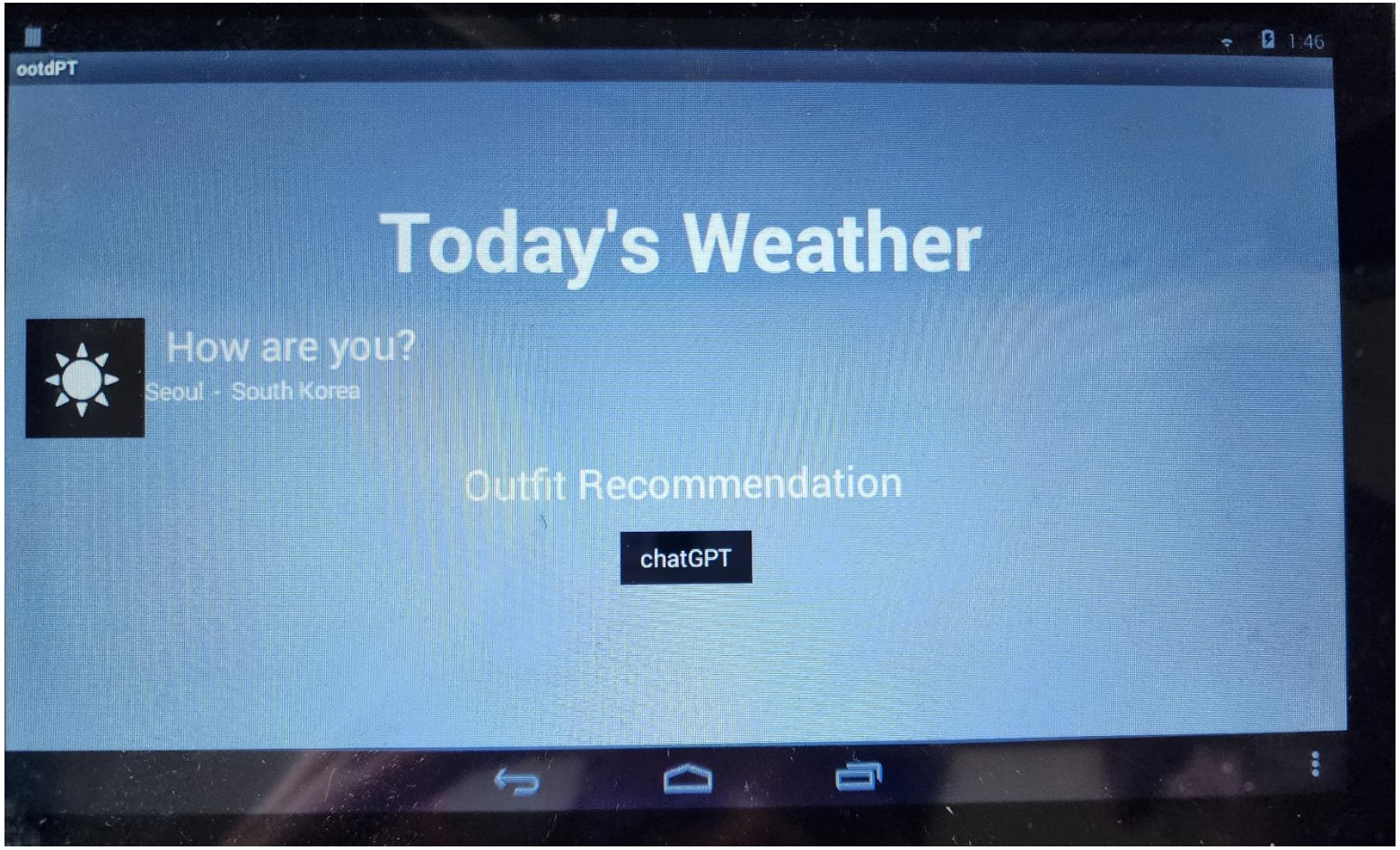
2 *Background Music Playing*





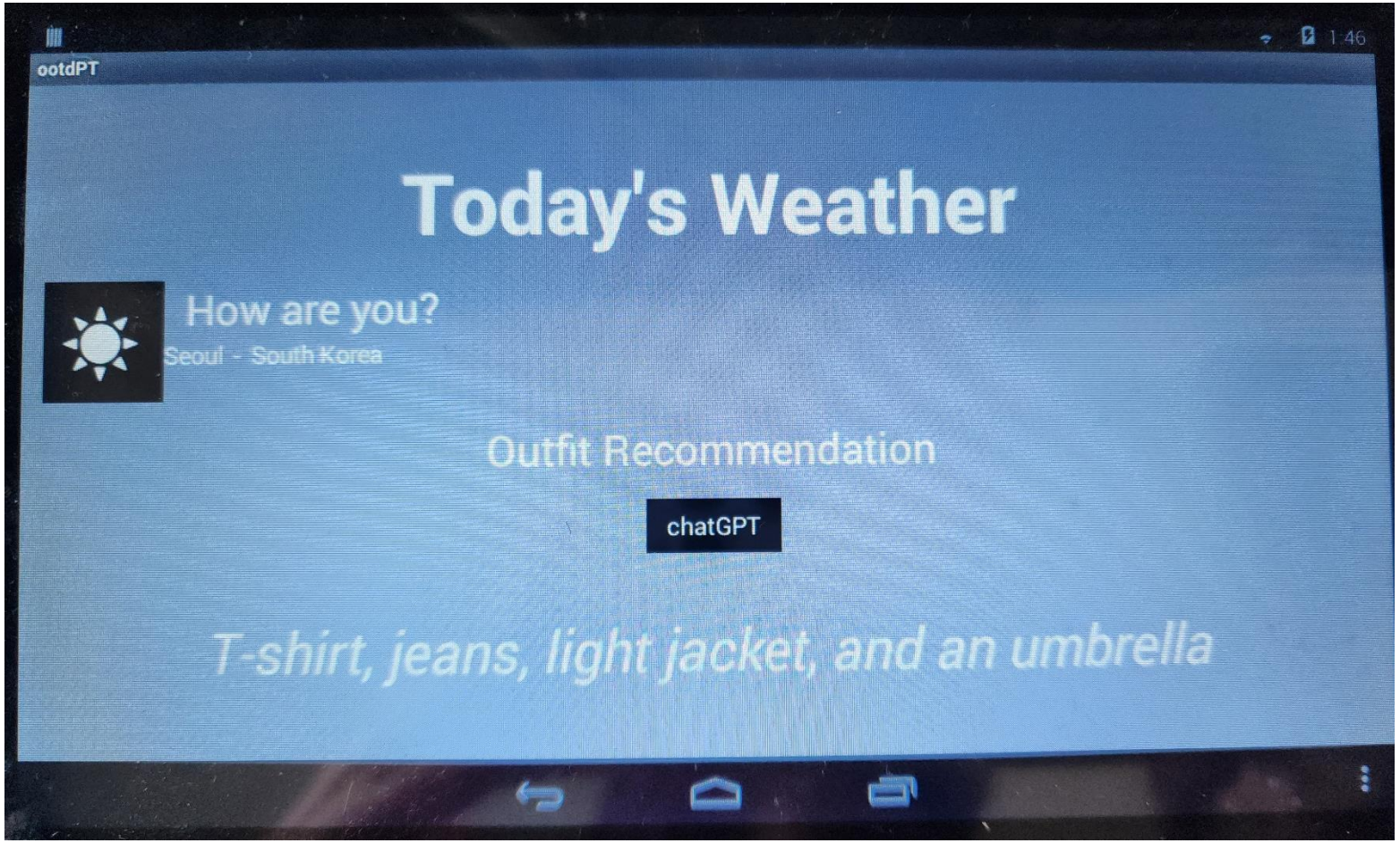
Turn on/off the music

Result - ootdPT



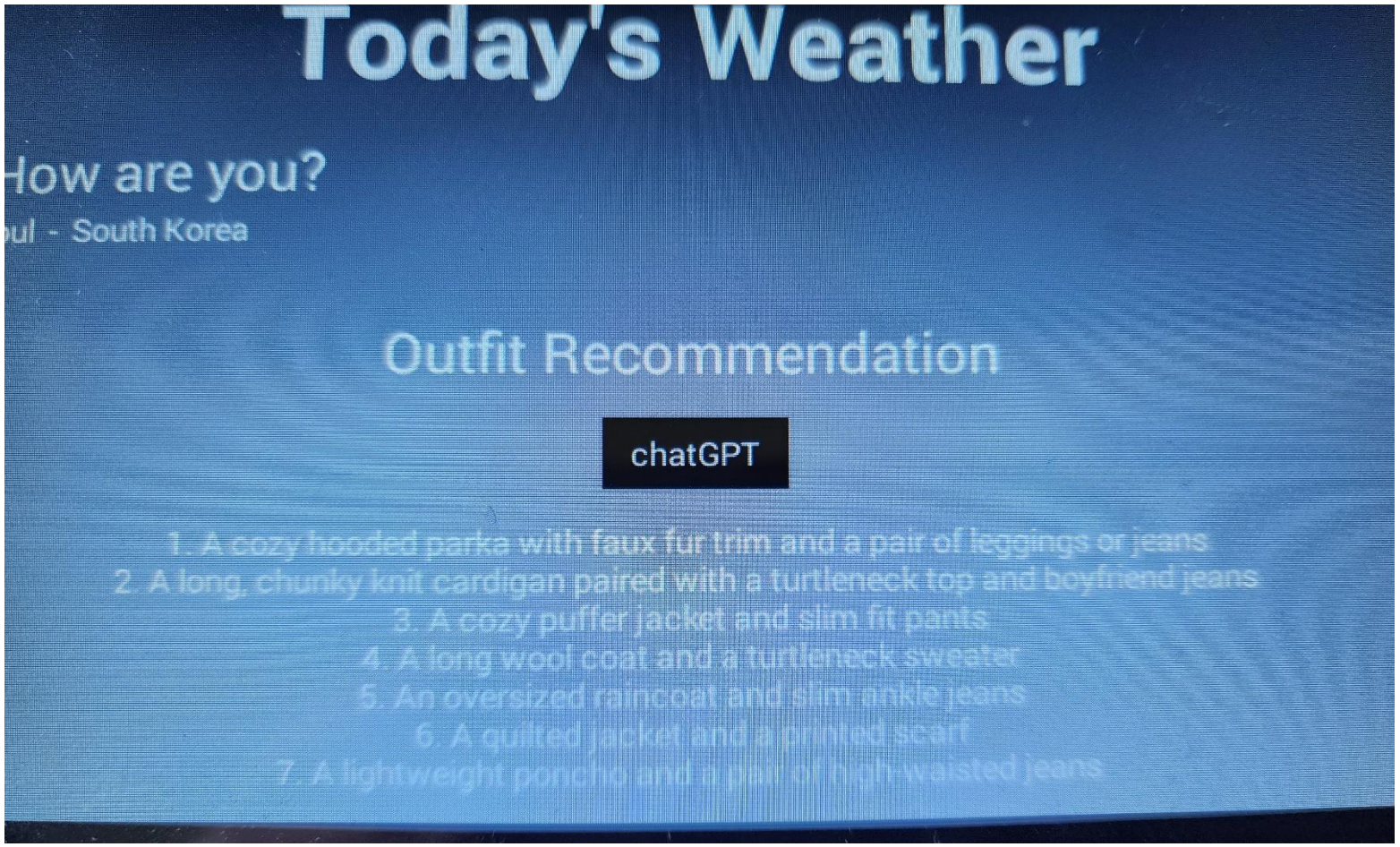
1. Wait for API callback
(takes under 1sec)

Result - ootdPT



2. In-device Engine works

Result - ootdPT



3. Get chatGPT Response

(if you click the button)

Let’s run it now!