



การทำ task schedule

คือการตั้งค่าให้โปรแกรมทำงานตามที่เรากำหนด



Task Schedule คือ ?

ฟังก์ชันที่เอาไว้กำหนดเวลาการทำงานของโปรแกรม เป็นการกำหนดให้โปรแกรมที่เราเขียนนั้นสามารถเริ่มทำงานได้ด้วยตนเองจากเวลาที่ได้กำหนดไว้ โปรแกรมจะเขียนคำสั่งตั้งเวลาด้วย cronjob ทำงาน



คำสั่งสร้าง task schedule

```
php artisan make:command HeatCron --command=heat:cron
```

ตำแหน่งไฟล์ที่ถูกสร้าง

```
app/Console/Commands
```



แก้ไขไฟล์ HeatCron.php

เพิ่มด้านบน

```
use App\Http\Controllers\HeatController;
```

และเรียกใช้งานที่ฟังก์ชัน handle()

```
$heat = new HeatController();
```

```
$heat->fetchDaily();
```



เพิ่มคำสั่งใน `Consoles\Kernal.php`

เพิ่มในฟังก์ชัน `schedule`

```
$schedule->command('heat:cron')->everyMinute();
```



คำสั่งทดสอบการทำงาน ด้วย php artisan

คำสั่งเรียกดูรายการทั้งหมด

```
php artisan schedule:list
```

คำสั่งทดสอบการทำงานของฟังก์ชันที่เราเขียนไว้

```
php artisan schedule:run
```



สั่งให้ `schedule` เริ่มทำงาน

คำสั่งที่ให้ `schedule` ทำงาน

```
php artisan schedule:work
```

โปรแกรมจะทำงานตามที่เรากำหนดค่าไว้

อ้างอิงจาก

<https://laravel.com/docs/8.x/scheduling#schedule-frequency-options>



การทำงานของ queues

queues คือ ตัวจัดการลำดับการทำงานในรูปแบบของคิว เพื่อให้โปรแกรมทำงานได้ในขณะที่มีการร้องขอเข้ามาจำนวนมาก ๆ อย่างเช่นการส่งอีเมลให้กับลูกค้าลักษณะแบบนี้จะใช้ให้ queue ทำงาน เพราะผู้ใช้งานไม่จำเป็นต้องรออีเมลขณะใช้งาน



เริ่มต้นสำหรับทำ queue

สร้าง migration สำหรับ queue คำสั่งคือ

```
php artisan queue:table
```

และคำสั่งสำหรับ migrate เข้าฐานข้อมูล

```
php artisan migrate
```

และทำการกำหนดค่าที่ .env สำหรับทำ queue เป็น

```
QUEUE_CONNECTION=database
```



คำสั่งในการสร้าง queue

คำสั่งสำหรับสร้าง HeatJob

```
php artisan make:job HeatJob
```

ตำแหน่งไฟล์

```
app\Jobs\HeatJob.php
```

กำหนดให้ HeatJob ทำงาน เมื่อถูกเรียก

เราจะกำหนด HeatJob ทำการ insert heat ที่ได้มาจากการดึงข้อมูลเข้าไปเก็บไว้ในตาราง Heat

```
protected $url = '';  
public function __construct($url)  
{  
    $this->url = $url;  
}
```

```
public function handle()  
{  
    $res = Http::get($this->url);  
    $data = $res->object();  
  
    if ($res->ok()) {  
        foreach($data->data->data as $val){  
            $heat = new Heat();  
            $heat->province = $val->province;  
            $heat->date = $val->d_date_utc;  
            $heat->maxTemperature = $val->maxTemperature;  
            $heat->save();  
        }  
    } else {  
        \Log::error($res->status());  
    }  
  
    return true;  
}
```



การเรียกใช้งาน HeatJob

เพิ่มด้านบน

```
use App\Jobs\HeatJob;
```

ให้ controller เป็นตัวเรียกใช้งาน HeatJob โดยผ่านค่าที่ฟังก์ชัน dispatch()

```
HeatJob::dispatch($val);
```



รันคำสั่ง Job

Job งานจะถูกเก็บไว้ที่ตาราง jobs และ failed_jobs จะเป็นคำสั่งที่เราได้กำหนดการทำงานเอาไว้

คำสั่งที่ให้ job ทำงานคือ

```
php artisan queue:work
```

อ้างอิงจาก

<https://laravel.com/docs/8.x/queues>



ดึงข้อมูลจาก google sheet

เราจะทำการดึงข้อมูลจาก google sheet แล้วนำไปเก็บไว้ในฐานข้อมูลที่เรากำหนดไว้



ขั้นตอนการติดตั้ง Package

รันคำสั่ง

```
composer require revolution/laravel-google-sheets
```

ต่อด้วย

```
php artisan vendor:publish --provider="PulktJalan\Google\GoogleServiceProvider" --tag="config"
```

ขั้นตอนการติดตั้ง Package

ต่อด้วยการกำหนดค่าที่ config/google.php

```
// config/google.php

// OAuth
'client_id'      => env('GOOGLE_CLIENT_ID', ''),
'client_secret' => env('GOOGLE_CLIENT_SECRET', ''),
'redirect_uri'   => env('GOOGLE_REDIRECT', ''),
'scopes'         => [\Google\Service\Sheets::DRIVE, \Google\Service\Sheets::SPREADSHEETS],
'access_type'   => 'online',
'approval_prompt' => 'auto',
'prompt'        => 'consent', // "none", "consent", "select_account" default:none

// or Service Account
'file'          => storage_path('credentials.json'),
'enable'        => env('GOOGLE_SERVICE_ENABLED', true),
```




เพิ่มตัวแปรใน .env

GOOGLE_APPLICATION_NAME=

GOOGLE_CLIENT_ID=

GOOGLE_CLIENT_SECRET=

GOOGLE_REDIRECT=

GOOGLE_DEVELOPER_KEY=

GOOGLE_SERVICE_ENABLED=

GOOGLE_SERVICE_ACCOUNT_JSON_LOCATION=

POST_SPREADSHEET=

POST_SHEET_ID=



ขั้นตอนการขอ google client id

เข้าไปที่ลิงค์ <https://console.cloud.google.com/>

1. คลิกที่ปุ่ม select-project เลือก new project จากนั้น กรอกข้อมูลที่จำเป็น กดบันทึก
2. คลิกที่เมนูด้านซ้าย credentials และเลือก create credentials ด้านบน
3. ทำการสร้าง API key
4. ทำการสร้าง OAuth Client ID
5. ทำการสร้าง Service Account
6. คลิกที่ Account ที่ได้สร้าง เลือกแท็บ KEY คลิก ADD KEY นำไฟล์ json ที่ได้ ไปวางที่ storage/
7. คลิกที่เมนู Library ค้นหา google sheet api และคลิก Enable เพื่อเปิดการใช้งาน
8. คลิกที่เมนู Library ค้นหา google drive api และคลิก Enable เพื่อเปิดการใช้งาน



ขั้นตอนการนำ Service Account ไปใช้งาน

เปิดที่ google sheet งานของเรา คลิกที่แชร์

เพิ่ม account ของ service account ที่ได้จาก google เข้าไป

กำหนดสิทธิ์ให้เป็น เอดิเตอร์



คำสั่งที่ใช้ในการดึงข้อมูล

เพิ่มด้านบน

```
use Revolution\Google\Sheets\Sheets;
```

ฟังก์ชันที่ใช้งาน

```
$sheets = new Sheets();  
$rows = $sheets  
    ->spreadsheet(config('sheets.post_spreadsheet_id'))  
    ->sheet(config('sheets.post_sheet_id'))  
    ->range('A1:A10')  
    ->all();
```