模組D - 排程管理系統

注意事項：為與國際競賽接軌，本次競賽不提供本地端XAMPP環境，選手需將成果傳送至遠端伺服器，確認作答成果。選手需熟悉使用SFTP/FTP將成果傳送到遠端這種開發模式。常用的工具如:FileZilla、IDE(PhpStorm...)內建遠端連線功能。

近期AI繪圖、大語言模型等技術的發展，這些技術的應用也越來越廣泛。但這些生成式AI生成內容時會需要大量的計算，導致使用者拿到的結果需要等待很久。這時需要一個系統進行任務的管理與分配，你將透過PHP框架來開發一個簡易的排程管理系統。

系統內建帳號：

#	Email	暱稱	密碼	身份
1	admin@web.tw	admin	adminpass	管理者
2	user1@web.tw	user1	user1pass	使用者
3	user2@web.tw	user2	user2pass	使用者
4	user3@web.tw	user3	user3pass	使用者

D.1-系統功能說明:

A. 角色

任務管理系統中有三種角色：管理員、使用者、Worker。除了登入登出外，各個角色不能使用其他角色的功能

管理員

1. 帳號密碼登入
2. 帳號登出
3. 使用者管理
    - 查看使用者列表
    - 修改使用者
4. 任務類型管理
    - 查看任務類型列表
    - 新增任務類型
    - 刪除任務類型
    - 執行結果統計
5. 任務管理
    - 查詢任務列表
6. 額度管理
    - 新增使用者額度
7. Worker管理
    - 新增Worker
    - 刪除Worker
    - 修改Worker

使用者

1. 帳號密碼登入
2. 帳號登出
3. 註冊帳號
4. 任務管理
    - 取得任務類型列表
    - 查詢任務列表
    - 新增任務
    - 取消任務
    - 查看指定任務資訊
5. 使用額度管理
    - 查詢使用額度
    - 查詢額度變更紀錄

Worker

1. 任務管理
    - 取得執行的任務
    - 回報任務執行結果

B. 登入登出功能說明

1. 使用帳號密碼登入後會獲得一個access token，access token為使用者email進行sha256，格式為hex格式，全小寫。
2. 透過Access token呼叫API時要將access token放在header的X-Authorization裡，格式為: Bearer {access_token}，範例: X-Authorization: Bearer 559aead08264d5795d39a8a08fdffd
3. 登出後需要將access token清除

C. 任務說明

1. 新增任務類型時須設定類型名稱與輸入欄位，類型名稱只允許小寫英文、數字、底線，且為唯一值
2. 輸入欄位可以有多個，每個欄位需要有欄位名稱與欄位類型，欄位類型可以為string、number、boolean
3. 新增任務時需要與任務類型設定的輸入欄位類型相同
4. 任務狀態分為~~四~~五個: pending、processing、finished、failed、canceled

D. Worker說明

1. 新增worker後會產生一個worker用的access token，worker會透過此access token驗證
2. 新增worker時可以設定此worker可以處理的任務類型，取得執行任務時需要依據此設定篩選
3. 當worker透過「取得執行的任務」API取得任務後，需要依據新增順序回傳一個"pending"狀態的任務，同時將此任務狀態改為"processing"
4. 如果此worker已擁有"processing"的任務，會回傳此worker處理中的任務
5. 當任務完成後worker會透過「回報任務執行結果」API回傳結果

E. 使用額度說明

1. 每建立一個使用者時會給予10的額度，每新增一次任務後會扣除1額度，如果額度歸零將無法新增任務
2. 管理者可以新增額度給指定的使用者
3. 資料庫只會紀錄額度的變動紀錄，需另外計算剩餘額度

F. 其他

1. 依據HTTP標準RFC 7230中的Header Fields規定，Header欄位名稱不區分大小寫
2. API中回傳的所有timestamp格式為"YYYY-MM-DDThh:mm:ss"，範例: "2024-01-01T23:59:01"
3. 需使用提供的資料庫，且不可對資料結構做出更改
4. 完整的URL應包含通訊協議、伺服器位址、檔案路徑、檔名等資訊，例如: http://www.w3.org/cms-uploads/Hero-illustrations/groups.svg
5. 資料庫儲存的密碼需要經過password_hash處理
6. 刪除「任務類型」和「worker」時需使用軟刪除，刪除後的任務類型名稱可以被其他任務類型使用
7. 軟刪除後該資料的deleted_at會從null改成刪除的時間
8. 「任務類型」或「worker」軟刪除後關聯任務的「任務類型」和「worker」欄位顯示null
9. 查詢類的API的total_count為此搜尋條件的所有數量
10. API評分時將使用自動測試輔助評分，測試過程會重新設置資料庫
11. 當有錯誤發生時需要以此結構回傳，message依據發生的錯誤回傳，以401 Unauthorized為例，response body回傳格式為：

```
{ "success": false, "message": "MSG_INVALID_TOKEN" }
```

D.2 - API

本API資訊僅供參考，實際API輸入 / 輸出規範及範例將於場地檢查日、競賽前或競賽時提供。

資料格式

User	**使用者**
id: Number	使用者id，唯一值
email: String	使用者的email，唯一值
nickname: String	使用者的暱稱
profile_image: String	使用者的頭像URL，必須為完整的url
type: String	使用者的類型. ADMIN或USER
access_token: String?	使用者的登入token，只有在登入API時顯示，其餘時候不得存在
created_at: String	建立的時間，格式為timestamp

TaskType	**任務類型**
id: Number	任務類型id，唯一值
name: String	任務名稱只允許小寫英文、數字、底線，唯一值
inputs: **TaskTypeInput[]**	**TaskTypeInput**陣列，數量為0個或0個以上，格式請參考 **TaskTypeInput**
created_at: String	建立的時間，格式為timestamp

TaskTypeInput	**任務類型輸入**
name: String	欄位名稱，每個任務類型中的唯一值
type: String	資料類型(string, number, boolean)

Task	**任務資訊**
id: Number	任務id，唯一值
status: String	狀態(pending, processing, finished, failed, canceled)
updated_at: String	更新的時間，格式為timestamp
created_at: String	建立的時間，格式為timestamp

TaskDetail	**任務詳細資訊**
id: Number	任務id，唯一值
type: **TaskType**	任務類型，格式請參考**TaskType**
user: **User**	擁有者，格式請參考**User**
worker: **WorkerDetail**	執行此任務的worker，如果~~狀態為pending~~沒有worker時回傳null

TaskDetail	**任務詳細資訊**
status: String	狀態(pending, processing, finished, failed, canceled)
result: String	結果，未完成時為null,必須為完整的URL
updated_at: String	更新的時間，格式為timestamp
created_at: String	建立的時間，格式為timestamp

WorkerDetail	**worker類型**
id: ~~String~~Number	worker id，唯一值
name: String	worker的名稱
types: TaskTypes[]	此worker可以執行的任務類型
is_idled: Boolean	是否為閒置的狀態.如果有任務執行中為true
created_at: String	建立的時間，格式為timestamp

Quota	**額度類型**
id: Number	quota id，唯一值
value: Number	異動的數量
reason: String	異動的原因
created_at: String	建立的時間，格式為timestamp

API. 1 使用者登入

使用者輸入正確的帳號密碼後需回傳access_token

POST /api/user/login	header: N/A

Request Body
{
"email": "email",
"password": "password"
}	

Response Body
{
"success": true,
"data": **User**
}	

API. 2 使用者登出

需登入，登出後需撤銷原access_token的登入權限

POST /api/user/logout	header: Authorization Bearer {admin/user token}

~~Request Body~~

```
{
  "email": "email",
  "password": "password"
}
```

Response Body

```
{
  "success": true,
  "data": ""
}
```

API. 3 使用者註冊

~~需登入，登出後需撤銷原access_token的登入權限~~

註冊使用者，密碼需加密後存入資料庫

POST /api/user/register	header: N/A

Request Body (Form Data)

email	使用者的email，唯一值
password	使用者的密碼，需加密後存入資料庫
nickname	使用者的暱稱~~，只允許jpg和png~~
profile_image	使用者的頭像，只允許jpg和png

Response Body

```
{
  "success": true,
  "data": User
}
```

API. 4 取得任務類型

取得所有任務類型

GET /api/task/type	header: Authorization Bearer {admin token}

Query parameters

order_by	排序依據，非必要 • created_at: 發布時間(預設)
order_type	排序方式，非必要 • asc: 升冪排序(預設) • desc: 降冪排序
page	頁數，非必要 預設值為1
page_size	每頁數量，非必要 預設值為10

Response Body

```
{
  "success": true,
  "data": {
    "total_count": 100,
    "posts": TaskType[]
  }
}
```

API. 5 新增任務類型

新增任務類型

POST /api/task/type	header: Authorization Bearer {admin token}

Request Body

```
{
  "name": String,
  "inputs": TaskTypeInput[]
}
```

Response Body

```
{
  "success": true,
  "data": TaskType
}
```

API. 6 新增任務

新增任務，inputs依據type設定的inputs決定，需要驗證資料類型是否正確舉例:

TaskType的inputs	**Request Body的inputs**

[
{
"name": "prompt",
"type": "string"
},
{
"name": "limit",
"type": "number"
},
]
	{
	"prompt": "Hellow World!",
	"limit": 120
	}

POST /api/task	header: Authorization Bearer {user token}

Request Body
{
"type": ~~String~~**Number**,
"inputs": **Object**
}	

Response Body
{
"success": true,
"data": **TaskDetail**
}	

API. 7 查詢任務列表

查詢任務列表，管理員可以查詢所有人的任務，使用者只能查詢自己擁有的任務

GET /api/task	header: Authorization Bearer {admin/user token}

Query parameters

order_by	排序依據，非必要 • created_at: 發布時間(預設) • updated_at: 更新時間
order_type	排序方式，非必要 • asc: 升冪排序 • desc: 降冪排序(預設)
page	頁數，非必要 預設值為1
page_size	每頁數量，非必要 預設值為10
status	依據任務狀態篩選，非必要 篩選多個狀態時使用逗號分隔，例如: ?status=pending,processing

Response Body

```
{
   "success": true,
   "data": {
      "total_count": 100,
      "posts": Task[]
   }
}
```

API. 8 取得指定任務資訊

依據任務id取得指定任務資訊，使用者只能查詢自己擁有的任務，如果查詢的是其他的任務會顯示「不存在的任務」的錯誤

GET /api/task/{task_id}	header: Authorization Bearer {user token}

Response Body

```
{
   "success": true,
   "data": TaskDetail
}
```

此API以後皆為新增的API

/* API 9 刪除 */

API. 10 刪除任務類型

刪除任務類型

DELETE /api/task/type/{typetype_id}	header: Authorization Bearer {admin token}

Response Body
{
"success": true,
"data": ""
}	

API. 11 新增使用者額度

新增使用者額度

POST /api/user/quota/{user_id}	header: Authorization Bearer {admin token}

Request Body
{
"value": Number,
}	

Response Body
{
"success": true,
"data": ""
}	

API. 12 新增worker

新增worker

POST /api/worker	header: Authorization Bearer {admin token}

Request Body
{
"name": String,
"tasktypelist": String(**TaskTypeid(以逗號分隔)**)
}	

Response Body
{
"success": true,
"data": **WorkerDetail**
}	

API. 13 修改worker

修改worker

PUT /api/worker/{worker_id}	header: Authorization Bearer {admin token}

Request Body

```
{
   "name": String,?
   "tasktypelist": String(TaskTypeid(以逗號分隔))?
}
```

Response Body

```
{
   "success": true,
   "data": WorkerDetail
}
```

API. 14 刪除worker

刪除worker

DELETE /api/worker/{worker_id}	header: Authorization Bearer {admin token}

Response Body

```
{
   "success": true,
   "data": ""
}
```

API. 15 查詢使用者列表

查詢使用者列表

GET /api/user	header: Authorization Bearer {admin token}

Response Body

```
{
   "success": true,
   "data": {
      "total_count": count,
      "users": User[]
   }
}
```

API. 16 修改使用者

修改使用者

PUT /api/user/{user_id}	header: Authorization Bearer {admin token}

Request Body
{
"email": String,?
"nickname": String,?
"password": String,?
}	

Response Body
{
"success": true,
"data": **User**
}	

API. 17 取消指定任務

取消指定任務

DELETE /api/task/cancel/{task_id}	header: Authorization Bearer {user token}

Response Body
{
"success": true,
"data": ""
}	

API. 18 查詢剩餘額度

查詢剩餘額度

GET /api/user/leftquota	header: Authorization Bearer {user token}

Response Body
{
"success": true,
"data": Number(leftquotacount)
}	

API. 19 查詢額度變更紀錄

查詢額度變更紀錄

GET /api/user/quota	header: Authorization Bearer {user token}

Response Body

```
{
   "success": true,
   "data": {
      "total_count": count,
      "quotas": Quota[]
   }
}
```

API. 20 取得需執行任務

需要依據新增順序回傳一個"pending"狀態的任務，同時將此任務狀態改為"processing"，如果此worker已擁有"processing"的任務，會回傳此worker處理中的任務

GET /api/worker/task	header: Authorization Bearer {worker token}

Response Body

```
{
   "success": true,
   "data": TaskDetail
}
```

API. 21 回報任務結果

回報任務結果

POST /api/worker/task/{task_id}	header: Authorization Bearer {worker token}

Request Body (Form Data)

status	狀態，必須為finished或failed
result	結果圖片，必須為jpg或png

Response Body

```
{
   "success": true,
   "data": TaskDetail
}
```

D.3 - 錯誤訊息列表

#	訊息	狀態碼	情境	適用API
1	MSG_INVALID_LOGIN	403	使用者不存在、帳密有誤	1
2	MSG_USER_EXISTS	409	使用者已存在	3
3	MSG_INVALID_ACCESS_TOKEN	401	無效的Token	2、4、5、6、7、8、10、11、12、13、14、15、16、17、18、19、20、21
4	MSG_PERMISSION_DENY	403	權限不足	4、5、6、8、10、11、12、13、14、15、16、17、18、19
5	MSG_MISSING_FIELD	400	缺少必要欄位	1、3、5、6、11、12、21
6	MSG_WRONG_DATA_TYPE	400	資料格式錯誤	1、3、4、5、6、7、11、12、13、16、21
7	MSG_IMAGE_CAN_NOT_PROCESS	400	圖片格式錯誤(非圖片檔)	3、21
8	MSG_TASK_NOT_EXISTS	404	不存在的任務	8、17、21
9	MSG_TASKTYPE_INPUT_NAME_EXISTS	409	任務類型input的名稱以存在	5
10	MSG_USER_QUOTA_IS_EMPTY	409	使用者點數不足	6
11	MSG_USER_NOT_EXISTS	404	不存在的使用者	11、16
12	MSG_NO_TASK_PENDING	404	無任務可領取	20
13	MSG_TASKTYPE_NOT_EXISTS	404	不存在的任務類型	6、10、12、13
14	MSG_WORKER_NOT_EXISTS	404	不存在的worker	13、14
15	MSG_TASKTYPE_TYPE_ERROR	400	task_type內值錯誤	6
16	MSG_TASKTYPE_NAME_EXISTS	409	task_type_name已存在	5
17	MSG_WORKER_NAME_EXISTS	409	worker name已存在	12、13
18	MSG_TASK_IS_END	400	任務已結束	17、21

D.4 - API Endpoints

#	功能	Method	URL	完成	評分
1	使用者登入	POST	api/user/login		
2	使用者登出	POST	api/user/logout		
3	使用者註冊	POST	/api/user/register		
4	取得任務類型	GET	/api/task/type		
5	新增任務類型	POST	/api/task/type		
6	新增任務	POST	/api/task		
7	查詢任務列表	GET	/api/task		
8	取得指定任務資訊	GET	/api/task/{task_id}		
9	已	刪	除	-	-
10	刪除任務類型	DELETE	/api/task/type/{typetype_id}		
11	新增使用者額度	POST	/api/user/quota/{user_id}		
12	新增worker	POST	/api/worker		
13	修改worker	PUT	/api/worker/{worker_id}		
14	刪除worker	DELETE	/api/worker/{worker_id}		
15	查詢使用者列表	GET	/api/user		
16	修改使用者	PUT	/api/user/{user_id}		
17	取消指定任務	DELETE	/api/task/cancel/{task_id}		
18	查詢剩餘額度	GET	/api/user/leftquota		
19	查詢額度變更紀錄	GET	/api/user/quota		
20	取得需執行任務	GET	/api/worker/task		
21	回報任務結果	POST	/api/worker/task/{task_id}		

D.5 - 選手注意事項

以下說明時用到XX代表選手個人的崗位編號，Y代表模組編號

- 將完成的結果存在網站根目錄，用XX_Module_Y作為資料夾名稱
- 請確認已經將最新成果上傳至伺服器XX_Module_Y上，並運作正常
- 資料庫名稱為 webXX_module_y，請匯入54_module_d.sql

D.6 - 評分注意事項

以下說明時用到XX代表選手個人的崗位編號，Y代表模組編號

- 請確認是否將首頁命名為適當的名稱，使得用瀏覽器開啟API功能，API不可經過轉址，以使用者登入的API為例，網址必須為

[http://cXX.web/XX_Module_Y/api/user/login] 或是

[http://cXX.web/XX_Module_Y/public/api/user/login]

請勾選V您網站的首頁網址，以及API網址的開頭

[ ] - http://cXX.web/XX_Module_Y/api/

[ ] - http://cXX.web/XX_Module_Y/public/api/

崗位編號: _____

簽名: _____

註記

SQL檔task/worker_id欄位可能有問題

- worker_id應該要可以為空值

?表資料不一定需要填寫

新增的資料以<u>底線</u>表示

刪除的資料以~~刪除線~~表示

可能有問題的資料以<u>雙底線</u>表示

54_national_module_d_v7.1.0.pdf