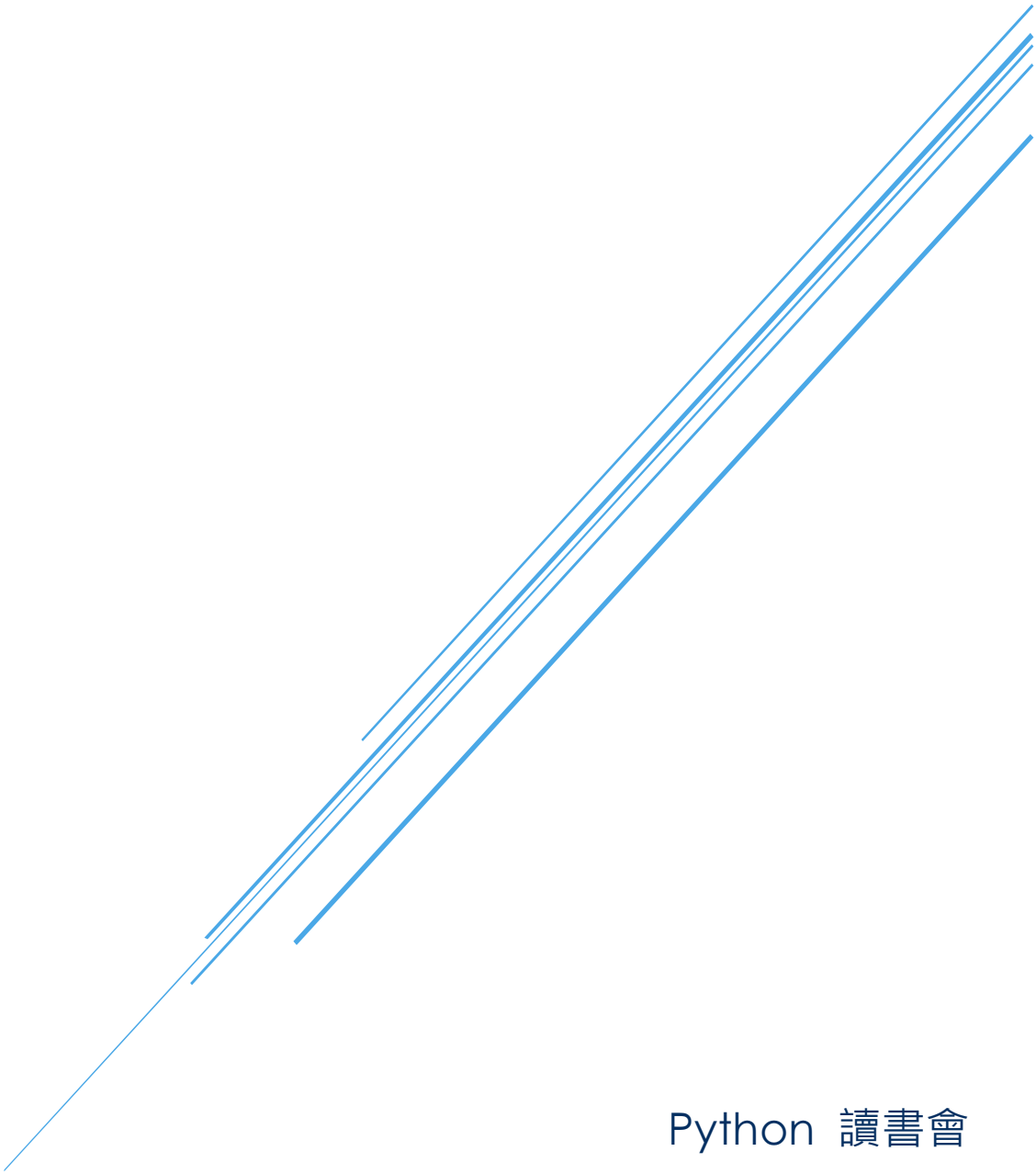
MANPOWER 計算系統

使用 JIRA Restful API 解決方案



Python 讀書會

Jerry & Van

版本紀錄

版本	內容	發佈日期	作者
v0.0.1	ManPower 軟體架構簡介	2024/09/17	Van

目錄

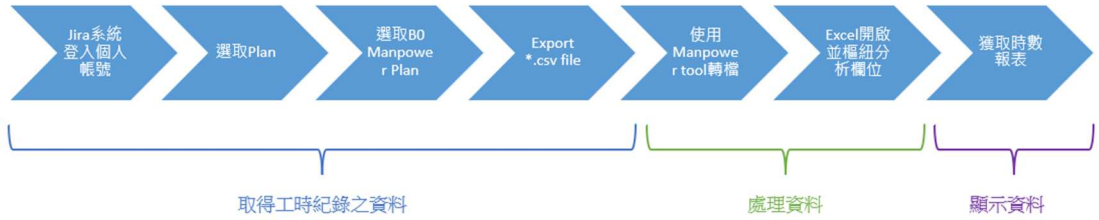
第 1 章　概論

第 1.1 節　MAN POWER 簡介

Man Power 是一套 AUO(友達光電有限公司)用來計算人力工時的簡易程式；主要方便觀察同仁每周工時，是否有**超過**或**不足**之情況，以便供同仁自行調整。該程式的運作流程如下：

從上面的流程架構圖來看，使用者必須**手動**進入 AUO 的 JIRA 系統，選擇自己的計畫(Plan)並且從 JIRA 中下載其 CSV 檔案(裡面有 BU 上所有成員在不同專案花費的時間)。下載完 CSV 檔案後，同仁便可以利用 Man Power Tool，將下載的 CSV 檔案轉換成 Excel 文件。最後，同仁便可以**手動**改成較容易閱讀的形式(如**樞紐分析**)，並且得知自己與別人每周的工時。

因此，不難發現這一連串的動作中，很多動作仍然需要使用者使用「**手動**」的方式，個別完成。從這點出發，我們可以重新分析一下整個流程，如下圖：



我們可以把整個流程分成三大部分，

(1) 取得工時紀錄之資料；

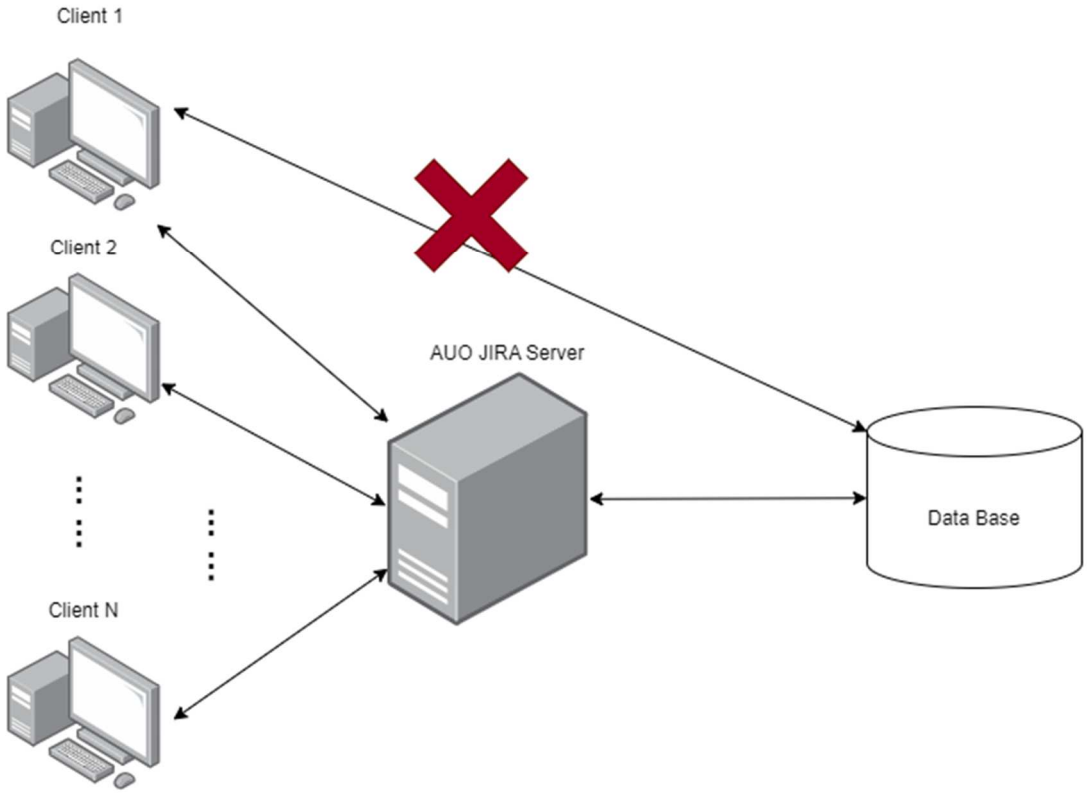
(2) 處理資料；

(3) 顯示資料；

而本次 Man Power Tool 工時統計計算程式的「**重點核心**」也在於「**取得工時紀錄之資料**」，我們會採用網際網路常用的技術「**HTTP**」協定與「**RESTful API**」。

第 1.2 節　軟體開發環境

要實現 Man Power Tool，我們首先必須要克服一個難題；那便是如何存取 AUO JIRA 伺服器，儲存在資料庫(Database)的資料呢？因為我們客戶端是無法直接取得 AUO JIRA 伺服器背後的資料庫伺服器，如下圖：



由於我們無法直接存取 AUO JIRA 伺服器的資料庫，因此只能採取「**間接**」存取的方式，進一步對 AUO JIRA 伺服器的資料庫進行更新以及讀取；也就是說，我們必須透過 AUO JIRA 伺服器，間接地更新以及取得資料庫的資料。

常見的方式有二，一是採用爬蟲的方式，透過剖析伺服器回應的網頁取得我們需要的數據，以及錄製網頁對於伺服器的請求，進而更新資料庫的數據；這個方法聽起來就非常的麻煩吧！另一個方法則是透過**伺服器提供**的 HTTP API 或者是 RESTful API 來間接存取 AUO JIRA 伺服器資料庫；而 HTTP API 和 RESTful API 的差異，在下一節會介紹。第二個方式，聽上去雖然簡單，但還是需要伺服器本身有提供，才能讓客戶端有辦法透過 API 來更新與取得資料庫的資料。

必須強調的是，不管是方法一還是二，兩者使用的技術都是基於 HTTP(HyperText Transfer Protocol)協定，或說兩種方式都是利用 HTTP 協定傳輸資料，只是採用的技巧有所不同。我們可以這樣理解，當伺服器本身沒有提供 HTTP API 時，我們便只能採用爬蟲的方式以及錄製網頁對伺服器發動 HTTP 請求去存取資料庫的資料。

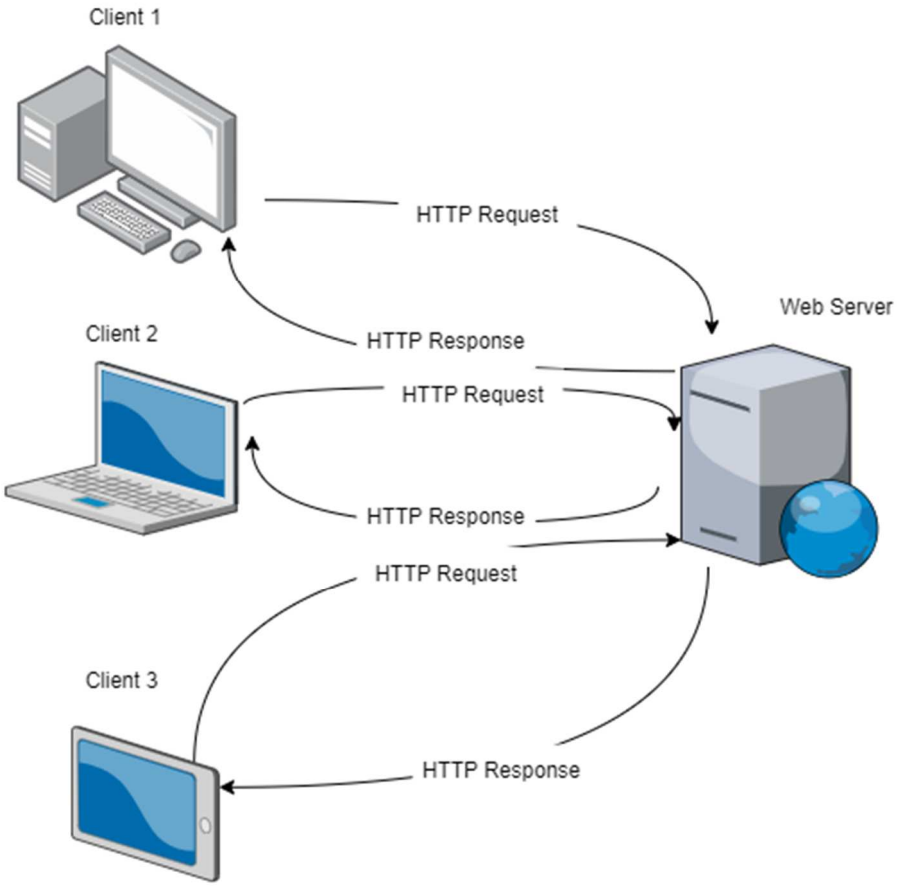
所幸，JIRA 伺服器有提供 HTTP RESTful API，我們可以利用 HTTP RESTful API 去存取 AUO JIRA 伺服器的資料庫。因此，在下一節，我們將簡介 HTTP 協定以及 RESTful API，然後進一步勾勒出我們軟體開發的環境。

第 1.3 節　HTTP　和　RESTFUL API

本節又分三小節，分別介紹 HTTP 協定、RESTful API 與 Man Power Tool 開發環境的介紹。

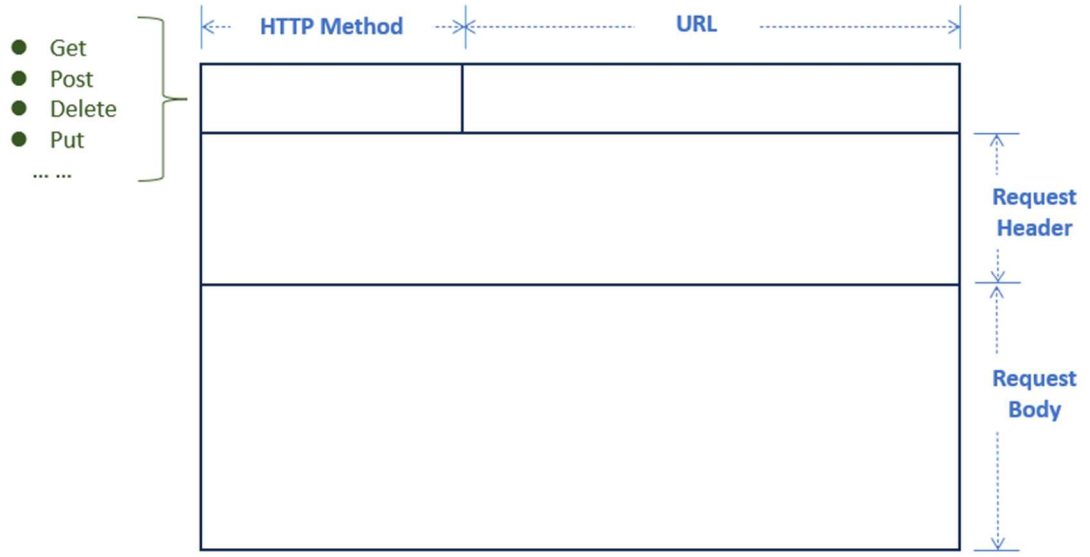
第 1.3.1 節　HTTP 協定

在網際網路的世界中，網站伺服器(Web Server)主要使用的通訊協定便是應用非常廣泛的**HTTP**(**H**yper**T**ext **T**ransfer **P**rotocol)協定，**HTTP** 協定採用**請求**(**Request**)/**回應**(**Response**)機制；**請求**的意思便是對網站伺服器請求「**資源**(**Resource**)」，而網站伺服器接收到該請求以後，如果該請求是伺服器允許的，便會給予對應的**回應**。如下圖：



HTTP 協定的誕生並不難以理解，畢竟網站伺服器需要與各式各樣不同地客戶端(Client)**溝通**，交換資源；為了避免溝通發生歧異，便需要統一的規範與通訊格式，HTTP 協定便定義了這一套客戶端與伺服器端之間溝通的方法。

既然 HTTP 協定如此重要，接下來，我們便需要簡單地了解一下 HTTP 請求與回應封包的格式；首先，我們先來認識一下 HTTP 請求封包，如下圖：



其中 HTTP Method 欄位常見的為 Get、Post、Delete 以及 Put，用來決定對網站伺服器進行的操作。URL 欄位則是填寫網站伺服器的位址。URL 的格式如下：

使用的協定://**域名**(or **IP**)[**:port**]/**路徑**(**path**)

例如：

http://192.168.1.223:8080/test

http://localhost:8080/test

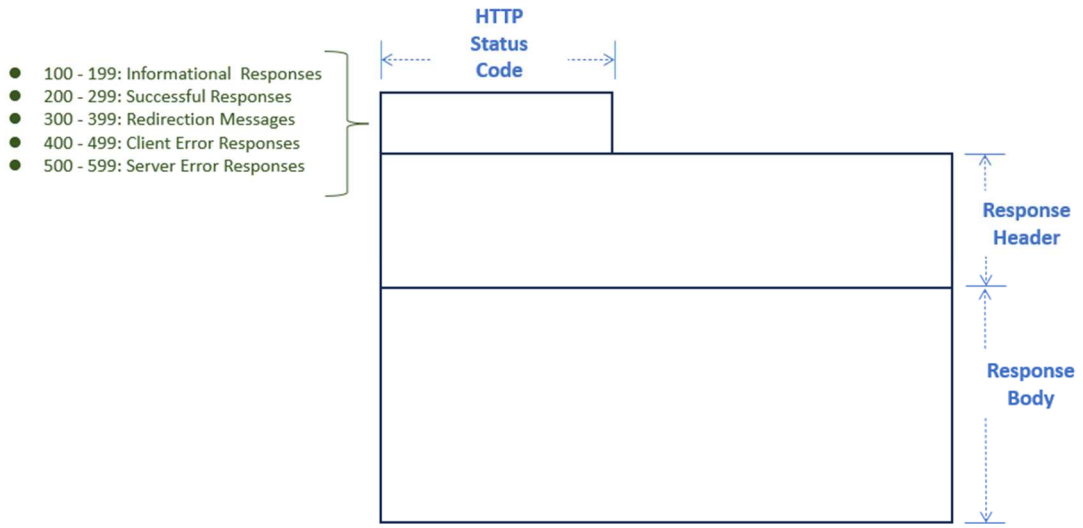
若沒有填寫埠號(Port)，預設為 80。Request Header 欄位可以讓使用者添加其額外的設定給予後端伺服器，如 Cookie。當 HTTP Method 為 Get 時，參數是放置在符號，"?"後面，例如：

http://localhost:8080/test?id=124&name=Judy

參數的格式為 Key=Value；多個參數時用"&"符號連接。若 HTTP Method 為 Put 或 Post 時，參數不會放置在 URL，而是放置在 Request Body 欄位之中，格式可以是 Key=Value 格式，

但若為 RESTful 風格，則為 XML 格式或是 JSON 格式。

HTTP Response 的封包格式則如下圖：



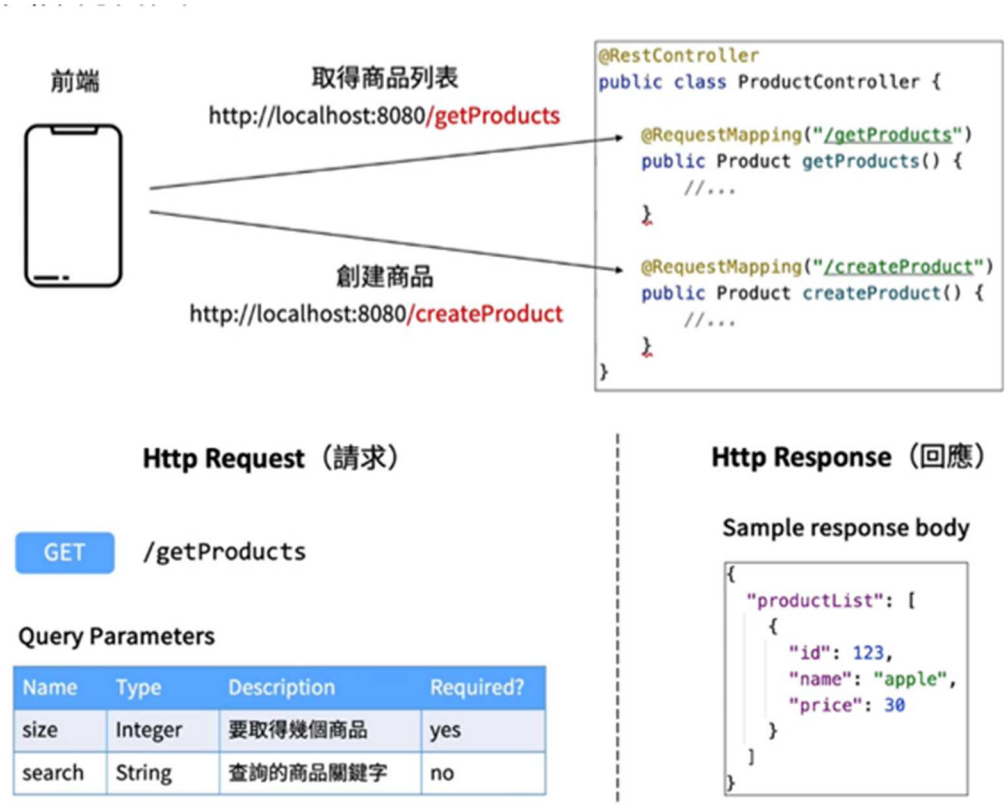
網頁伺服器收到請求後，會回應「請求的結果」，最直接明瞭的就是透過 HTTP Status Code 回應。我們不詳細介紹這個 HTTP Response，因為畢竟這些回應與實際的伺服器實作有很大的不同。更多有關 HTTP 的知識可以自行去網路搜索，我們就只將 HTTP 協定的基礎介紹到這裡。

第 1.3.2 節 RESTFUL API

在介紹 **RESTful API** 之前，我們先得解釋一下什麼是 **API**，所謂的 API 指的是：

1. API 是指用工程師的方式去說明某個功能的使用方法；
2. 目的：寫清楚這個功能要怎麼用；

而 **HTTP API**，自然就是要告訴開發前端或客戶端的工程師，要如何發送 HTTP 請求才可以得到或說達到伺服器提供預期的服務或是資源。如下頁的圖，



上圖提供 GET /getProducts，便代表前端或客戶端只需要發送 HTTP Method 為 GET，URL 路徑為 getProducts，且必須在"?"後面帶 size 參數的 HTTP 請求，便可以取得指定 size 的產品資訊。而所謂的 RESTful API 便是符合 REST 風格的 HTTP API。

何謂 **REST(Representational State Transfer)**？按照英文的翻譯，即**表現層狀態轉換**；這是甚麼意思呢？在網際網路的世界中，所有的一切統稱叫作「**資源**」(Resource)，而所謂的表現層狀態，指的便是「資源」的狀態。那麼如何表現出資源的狀態呢? 一種直觀的想法便是利用 HTTP Method 去表達、表現現在想要如何處理「資源」。如下表：

HTTP Method	對應的資料庫操作	說明
POST	Create (新增)	新增一個資源
GET	Read (查詢)	取得一個資源
PUT	Update (修改)	更新一個已經存在的資源
DELETE	Delete (刪除)	刪除一個資源

如上表，我們便可以使用 HTTP Method 的 POST、GET、PUT 以及 DELETE 去**表現**請求資源的**狀態**，要新增還是要刪除等等之**轉換**。

但是還有個問題，我們要對什麼「資源」進行處理呢？為了讓 HTTP 請求可以**表現**出來這個狀態，**REST** 風格採用了 URL 路徑來描述資源之間的關係，如下表範例：

Http method + URL 路徑	說明
GET/users	取得所有 user
GET/users/123	取得 user id 為 123 的 user
GET/users/123/articles	取得 user id 為 123 的 user 所寫的所有文章
GET/users/123/articles/456	取得 user id 為 123 的 user 所寫的、article id 為 456 的文章
GET/users/123/videos	取得 user id 為 123 的 user 所錄的所有影片
GET/users/123/videos/789	取得 user id 為 123 的 user 所錄的、video id 為 789 的影片
GET/users/100	取得 user id 為 100 的 user

最後，則是傳輸資料之間的格式能夠統一，因此 REST 風格要求資料必須以 XML 格式或是 JSON 格式。

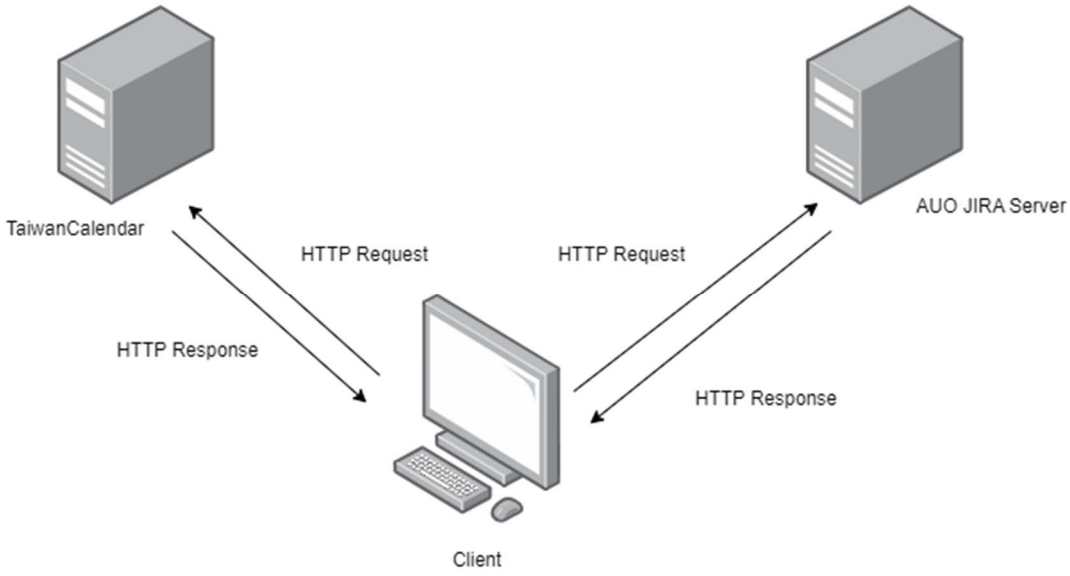
總結一下 RESTful API 的準則如下：

1. 使用 **HTTP** method 表示動作；
2. 使用 **URL** 路徑描述資源之間的階層關係；
3. response body 返回 json 或是 xml 格式；

RESTful API 沒有標準答案，永遠記住 RESTful API 實現的目的便是為了簡化團隊溝通的成本，我們本小節的介紹旨在助於我們能更好理解 JIRA API 設計，更多 RESTful API 的知識還是可以透過各式各樣的網路文件得到更進一步的認識。

第 1.3.3 節 MAN POWER TOOL 環境

Man Power Tool 的環境如下圖：



由於我們需要取得台灣的行事曆，所以 Man Power Tool 所在的客戶端，除了需要向 AUO JIRA 伺服器取得與更新資料外，也需要向 Taiwan Calendar 的伺服器取得目前年份的行事曆。

第 2 章　MAN POWER TOOL 軟體需求

第 3 章 MAN POWER TOOL 軟體架構簡介

根據前兩章的介紹，我們將在本章中架構一個軟體結構去完成軟體需求。在第 1 章中，我們分析了整個 Man Power Tool 系統，可以分為：

(1) 取得工時紀錄之資料；

(2) 處理資料；

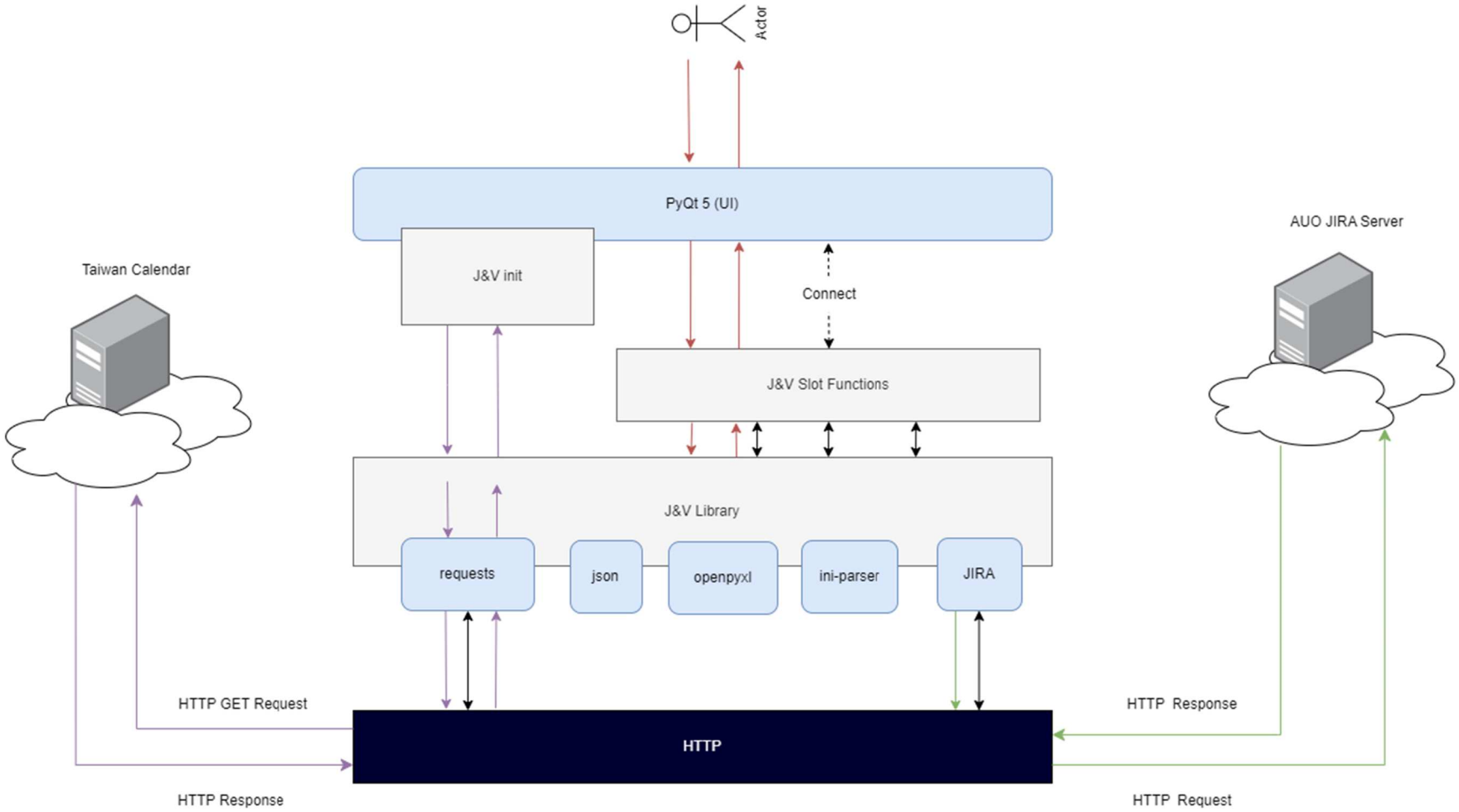
(3) 顯示資料；

同時我們也介紹了我們如何使用 RESTful API 去「取得工時紀錄之資料」。而在第 2 章，我們也介紹我們「處理資料」和「顯示資料」的需求。而在本章中，我們會把這些需求以具體的方式呈現出來。

第 3.1 軟體設計架構

由於空間不足，我們將 Man Power Tool 的軟體設計架構圖放置在下一頁，如下頁圖，不難發現 Man Power Tool 的軟體架構是由一些淺藍色方塊、一些淺灰白色方塊以及一個黑色方塊「HTTP」所構成；這裡，值得我們注意的只有圖中的淺藍色方塊以及淺灰白色方塊；淺藍色方塊為 Python 開放原始碼的模組，而淺灰白色方塊為我們自行封裝的模組，主要的目的是橋接使用者介面、Man Power Tool 的運作演算法，以及封裝 Python 開放原始碼模組。

首先，我們會利用 Python 的 PyQt5 套件打造 Man Power Tool 使用者介面(User Interface, UI)，然後在 UI 與 Python 開放原始碼模組之前，隔一層由我們所撰寫的程式模組去銜接這兩層。如此，Man Power Tool 的軟體架構便組建完成。更多的內容，我們會在之後的章節做更完整的介紹。


Actor
PyQt 5 (UI)
J&V init
Connect
J&V Slot Functions
J&V Library
requests
json
openpyxl
ini-parser
JIRA
HTTP
Taiwan Calendar
AUO JIRA Server
HTTP GET Request
HTTP Response
HTTP Response
HTTP Request

第 3.2 節　軟體套件簡介

本節延續上節，針對架構圖中淺藍色方塊的 Python 開放原始碼套件做一個簡介。

第 3.2.1 節　PYTHON PYQT5 套件

PyQt5 套件是基於 Qt 5 的 Python 開發套件；換言之，我們可以利用該套件的 Qt Designer 開發使用者介面，而後轉成 Python 程式碼。轉換後，便可以將之整合至我們的 Man Power Tool 之中。

第 3.2.2 節　PYTHON JSON 套件

由於大多數網站伺服器，特別是符合 RESTful API 的網站伺服器，伺服器回應的 HTTP Response 的資料格式大多都是 JSON 格式。因此為了方便開發，我們引入了這個 Json 套件，讓我們方便處理 JSON 字串。

第 3.2.3 節　PYTHON REQUESTS 套件

由於我們必須對網站伺服器發起 HTTP 請求，因此需要這個 Requests 套件去發送 HTTP API 或是 HTTP RESTful API 對網站伺服器進行增(Create)、查(Read)、改(Update)、刪(Delete) ，即 CRUD。

第 3.2.4 節　PYTHON INI-PARSER 套件

關於程式的一些預設值，或是設定值，為了能夠實現更佳彈性的設計，我們通常會把這些設定值獨立於程式之外，成為設定檔案。最常見的設定檔案便為 ini 設定檔案，而 ini-parser 套件便是用來剖析這類設定檔案的套件。

第 3.2.5 節　PYTHON JIRA 套件

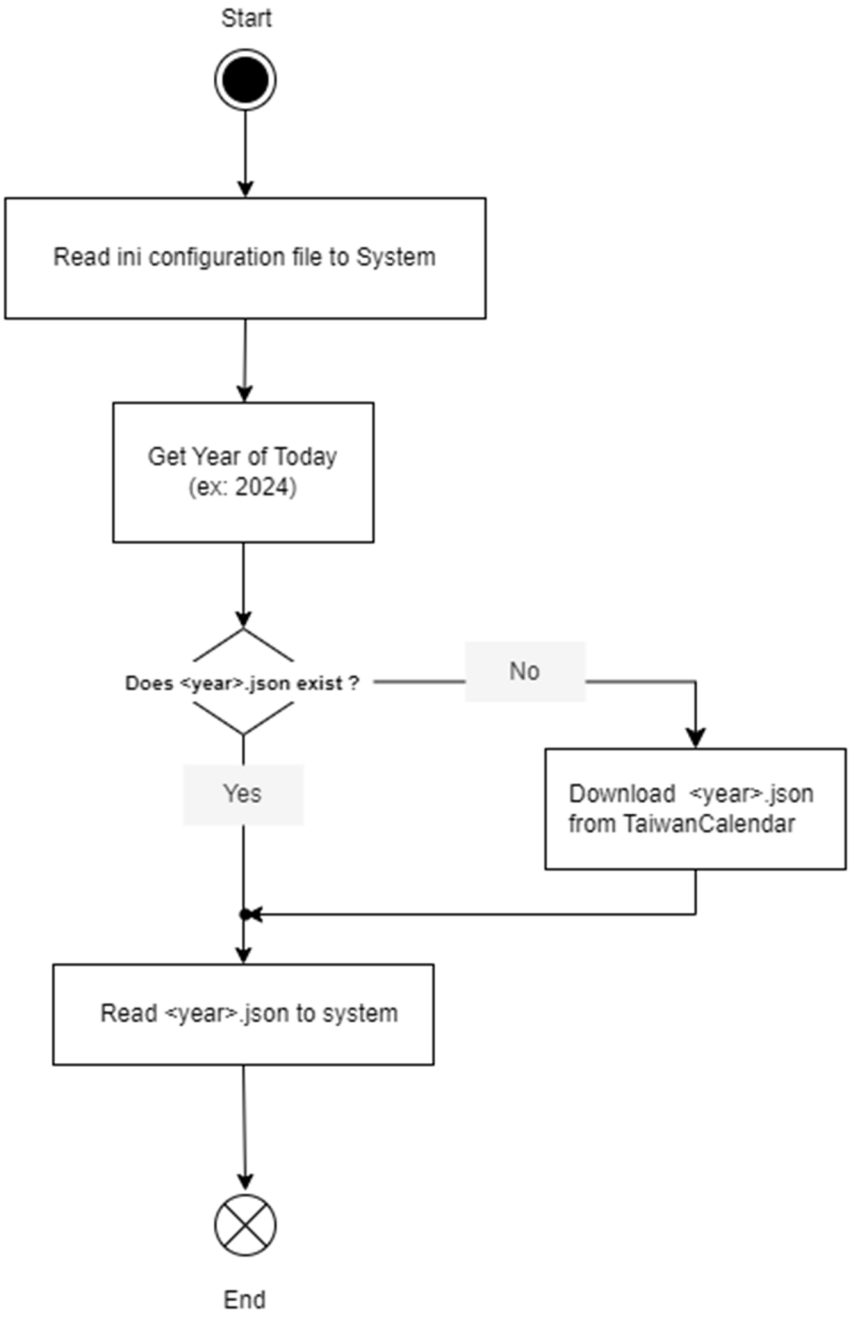
除了使用 Requests 套件可以呼叫 HTTP API 以及 HTTP RESTful API，針對 JIRA 的伺服器，Python 也有專門的 jira 套件可以對 JIRA 伺服器進行操作。

第 3.2.6 節　PYTHON OPENPYXL 套件

除了取得以及處理資料外，Man Power Tool 還需要將其整理成 EXCEL，openpyxl 便是方便我們操作 EXCEL 的套件。

第 3.3 節 MAN POWER TOOL 初始化執行流程

這一節將介紹 Man Power Tool 初始化執行流程；當 Man Power Tool 程式被使用者點擊後，程式便會開始運作，而在運作一開始便會執行初始化之動作。如下圖：



首先程式一開始會在指定目錄下讀取 ini 設定檔案，並且把這些設定從檔案載入至記憶體；而後系統需要取得今年台灣的行事曆，該行事曆可以至如下網址下載：

https://github.com/ruyut/TaiwanCalendar

因此我們必須取得今天日期的年份，並且根據年份至該網站取得以年份命名的 JSON 檔案，以今年 2024 年為例，便需要取得 2024.json 檔案。但問題來了，這個檔案理論上應該只需要下載一次便可，而不需要每次 Man Power Tool 被點擊都被執行一次才對；於是我們在取得今年的年份後，便可以至指定的位置去判斷<year>.json 的檔案是否存在？

如果該檔案不存在，便要至 TaiwanCalendar 網站下載<year>.json 檔案；而如果檔案存在，那便讀取<year>.json 檔案，並且選擇一個適當的資料構將之載入記憶體，以便後續程式演算法的使用與呈現。