Lab10 Web & Database

提敘:

實作一個簡化的 RESTful API 的 web server，允許新增、讀取、更新及刪除 key-value pair 等 CRUD 操作，並設定 Nginx 作為 web server (or reverse proxy)。Judge 會由 http://<wan_ip> 進行測試。
若沒有任何 web 開發經驗，後端建議使用:

- Python: FastAPI, Flask
- Node.js: Express
- PHP

資料庫則建議使用常見的 MySQL、SQLite 等等。

目標:

實作下列五個 APIs，本次 Lab 中資料的傳輸都用 JSON 格式:

1. `POST /key`: 新增一個 key-value pair
    a. Request body 須包含 key 和 value 的值，Ex:

```
{
  "key": "key1",
  "value": "data1"
}
```

    b. 資源成功創建須回傳 201，若 key 已存在則回傳 400 (key 須為唯一)。
2. `GET /key`: 回傳 database 中所有的 key (不用回傳 value)
    a. Response 為 array，包含 database 中所有的 key，Ex:

```
[
  "key1",
  "key2"
]
```

    b. 回傳 200 狀態
3. `GET /key/{key}`: 回傳 database 中指定的 key 的資訊
    a. Request，Ex: `curl http://<YOUR_IP>/key/key1`
    b. Response，Ex:

```
{
  "key1": "data1"
}
```

    c. 成功則回傳 200，若找不到對應的 key 就回傳 404
4. `PUT /key/{key}`: 更新指定的 key 的 value
    a. Request body 須包含 value 的值，Ex:

```
{
  "value": "updated_data1"
}
```

    b. 因為是 PUT，所以 key 不存在時會創建新的 key-value pair，請回傳 201。若 key 已存在，則更新 value 並回傳 200。
5. `DELETE /key/{key}`: 刪除指定的 key
    a. Request，Ex: `curl -X DELETE http://<YOUR_IP>/key/key1`
    b. 回傳 200

請設定 Nginx 作為 web server (or reverse proxy)，並將 Nginx 服務開在 80 port，資料的儲存則不限制任何形式，但我們建議使用 database。

注意事項:

1. Judge 前請確保資料庫清空，否則 judge 判定可能會失效。註：若已成功實作好 `GET /key` 和 `DELETE /key/{key}`，我們會幫你清空資料庫的資料。
2. 資料傳輸都以 JSON 格式傳送。
3. 由於 key 會包含任何字元，請記得做 URL encode。
4. Key 和 value 的最大長度為 200 個字元。

測試指令:

Test at external or internal	Host	Target	Test command	Score
External	VM	確認 80 的服務是否為 Nginx 以及檢查 Server header 是否為 nginx	pid=$(sudo ss -antlp | grep 0.0.0.0:80 | awk -F ',' '{print $2}' \| cut -d '=' -f 2) sudo cat /proc/$pid/cmdline | grep -ao nginx # 輸出應為 nginx curl -Ssi http://localhost | grep Server: | grep -o nginx # 輸出應為 nginx	10
		嘗試 POST 新增 key-value pair 並檢查回傳的狀態碼，接著 GET /key/{key} 確認是否存在	# judge 過長，不列出	20
		檢查前一個測試中新增的 key，是否能在 GET /key 這個 API 中搜尋到	# judge 過長，不列出	20
		確認是否能透過 DELETE 將創建的 key 刪除	# judge 過長，不列出	20
		確認當 key 不存在時，GET /key/{key} 回傳 404	# judge 過長，不列出	10
		檢查 PUT，以及其回傳的狀態碼	# judge 過長，不列出	20

Lab Description:

Implement a RESTful API web server, which allows clients to create, read, update and delete a key-value pair. In addition, configure Nginx as web server (or reverse proxy).

If you are not experienced in web development, we suggest you learning one of the backend frameworks or languages listed below to complete this lab:

- Python: FastAPI, Flask
- Node.js: Express
- PHP

As for the database, we suggest MySQL or SQLite.

Requirements:

Implement the APIs listed below. The data is transmitted using JSON format:

1. `POST /key`: Create a key-value pair
    a. Request body should contain the value of “key” and “value” fields, Ex:

```
{
  "key": "key1",
  "value": "data1"
}
```

    b. Return 201 status if the resource is created. If the key exists, return 400 (“key” must be unique).
2. `GET /key`: List “key” field for all key-value pairs in the database (no need to return “value” field)
    a. Response is an array, which includes “key” field for all key-value pairs, Ex:

```
[
  "key1",
  "key2"
]
```

    b. Return 200 status
3. `GET /key/{key}`: Return the information for the specified key
    a. Request Ex: `curl http://<YOUR_IP>/key/key1`
    b. Response Ex:

```
{
  "key1": "data1"
}
```

    c. Return 200 status if the request succeeds. Return 404 if the specified key cannot be found.
4. `PUT /key/{key}`: Update “value” field of the specified key
    a. Request body should contain the value of “value” field, Ex:

```
{
  "value": "updated_data1"
}
```

    b. Because it is using PUT method, the key-value pair should be created if it is not found. You should return 201 status in this case. If the specified key exists, return 200 after updating the value.
5. `DELETE /key/{key}`: Delete the specified key
    a. Request, Ex: `curl -X DELETE`

`http://<YOUR_IP>/key/key1`

b. Return 200 status

Configure Nginx as web server (or reverse proxy) and listen at port 80.

Precautions:

1. Please make sure that the database is empty before judging. Otherwise, the judge might be correct. Note that if you have implemented `GET /key` and `DELETE /key/{key}` APIs, we will clear the data for you.
2. Data is transmitted using JSON format.
3. Because the key might contain any characters, you should URL encode the key.
4. The lengths of key and value are less than 200 characters.

Tests:

Test at external or internal	Host	Target	Test command	Score
External	VM	Check if Nginx is listening at port 80 and the server header is nginx	pid=$(sudo ss -antlp | grep 0.0.0.0:80 | awk -F ',' '{print $2}' \| cut -d '=' -f 2) sudo cat /proc/$pid/cmdline | grep -ao nginx # This should output nginx curl -Ssi http://localhost | grep Server: | grep -o nginx # This should output nginx	10
		Try creating key-value pairs through POST API and check the return status. Next, check if the key exists through GET /key/{key} API.	# TL;DR	20
		Check if the newly created keys are listed in GET /key API.	# TL;DR	20
		Check if the created key can be deleted through DELETE API.	# TL;DR	20
		When the key doesn't exist, GET /key/{key} should return 404 status	# TL;DR	10
		Check PUT API and its return status	# TL;DR	20