Lab 8
RESTful API

CSCI2720 Building Web Applications

Agenda

- Being RESTful
- Using HTTP for CRUD
- Setting up an API service

Web API

- Web API (Application Programmatic Interface) can provide
  - Data resources
    - *e.g.* bus arrival time, restaurant ratings, …
  - Services or ***microservices***
    - *e.g.* converting coordinates into place names, creating QR codes
- Developers can then easily incorporate these building blocks into other web applications

Being RESTful

- RESTful API is a common architecture
- To prepare a RESTful web service API, these are generally required:
  1. Use HTTP methods explicitly
  2. Be stateless
  3. Expose directory structure-like URIs
  4. Transfer data in JSON or XML

Using HTTP for CRUD

- HTTP Methods ←→ ***CRUD*** operations
  - **POST** – To *__C__*reate a resource on the server
  - **GET** – To *__R__*etrieve a resource
    - Idempotent, should not initiate a state change
    - Cacheable
  - **PUT** – To *__U__*pdate a resource
  - **DELETE** – To *__D__*elete a resource
- ***Note***: Though not recommended, it is possible to use only **GET** and **POST** methods to support CRUD operations

Being stateless

- Not to store session data in a local storage
  - *e.g.*, memory, disk
- Each request is complete and independent
- Being stateless promotes scalability
- ***Idempotence***: repeated calls produce the same result

Designing proper URI

- Every service is treated as a resource identifiable by a URI, *e.g.*
  - URI for a forum service of topics

    `http://www.myservice.org/discussion/topics/{topic}`
- Keep URI *consistent*, *straightforward*, *predictable*, and easily *understood*
- Keep URI ***hierarchical*** but not merely slash-delimited strings

Outputting data

- ***XML***
  - Standardized format
  - Good for strongly typed data
- ***JSON*** *(compared to XML)*
  - Lightweight
  - Easier to parse
- Give client applications the ability to request for a specific content type
  - Make use of the built-in HTTP **`Accept`** header

An example pattern

- List container contents: **`GET /items`**
- Add an item to container: **`POST /items`**
  - with item details in request body
  - URI of item returned in HTTP response header, *e.g.*

    `Location: http://host/items/itemid`
- Retrieve an item: **`GET /items/itemid`**
- Update an item: **`PUT /items/itemid`**
  - with updated item in request body
- Delete an item: **`DELETE /items/itemid`**

Your task…

- Using Node.js and Express on CSCI2720 VM, prepare these ***endpoints***:
  - List all courses:
    `GET` `/courses`
  - List details of one course:
    `GET` `/courses/`*`courseid`*
  - Add one course:
    `POST` `/courses`
  - Update one course:
    `PUT` `/courses/`*`courseid`*
  - Delete one course:
    `DELETE` `/courses/`*`courseid`*

Your task…

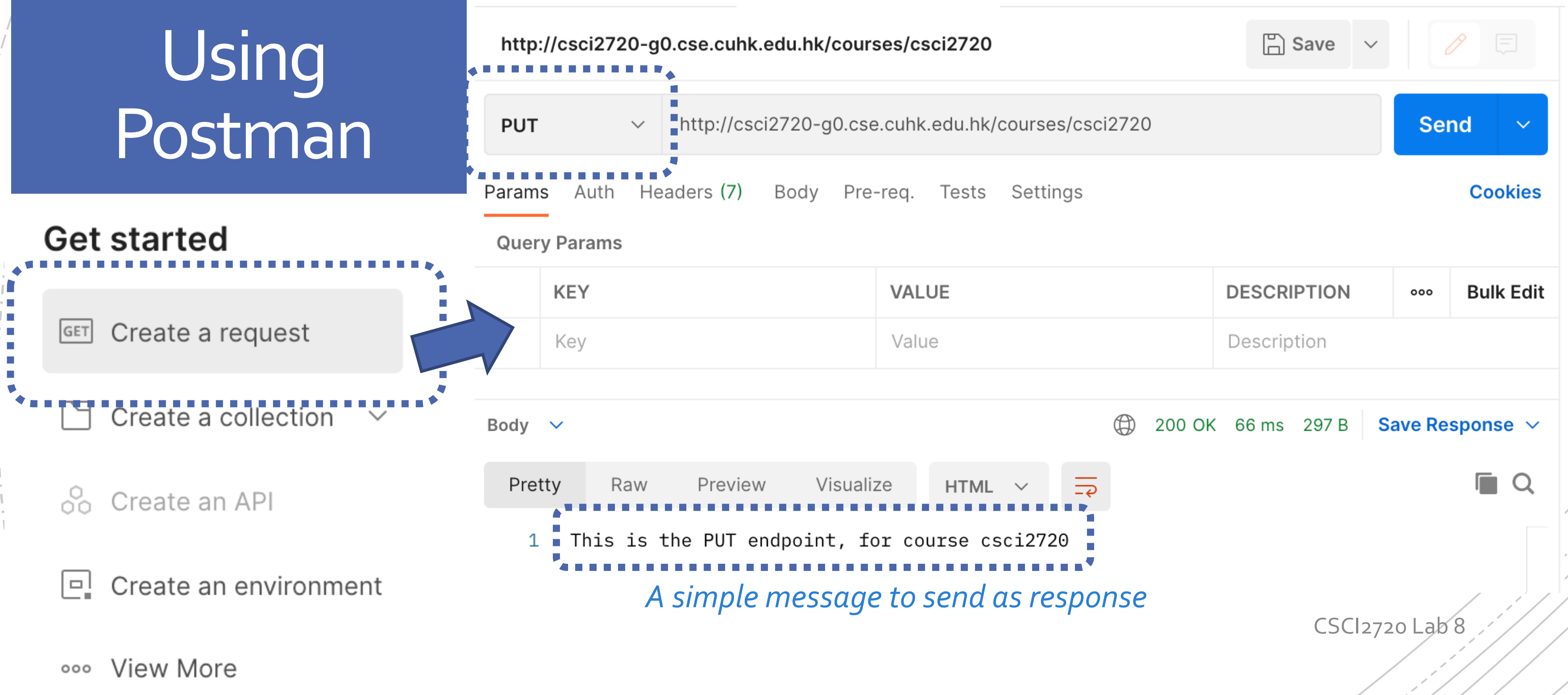
- For this lab, simply don't care about database or other data structure
- Just build the endpoints (URL to access) in Node using `app.get()`, …
- Output a simple message using res.send() for each endpoint, including `courseid` if there is

Using Postman

- While GET is easily achieved using a browser, POST, PUT and DELETE require extra effort
- Try using ***Postman*** to send appropriate HTTP requests
  - Download here: *https://www.postman.com/downloads/*
- Another possibility: the command line tool cURL

Using Postman

- *Note: You don't need any account or signing in to use this app!*
- Send requests by adjusting **URL**, **HTTP method**, and **body** (if needed)



Get started

Create a request

Create a collection

Create an API

Create an environment

View More

http://csci2720-g0.cse.cuhk.edu.hk/courses/csci2720

PUT http://csci2720-g0.cse.cuhk.edu.hk/courses/csci2720

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body 200 OK 66 ms 297 B Save Response

Pretty Raw Preview Visualize HTML

```
This is the PUT endpoint, for course csci2720
```

A simple message to send as response

Submission

- No submission is needed for labs
- What you have done could be useful for your further exploration or the upcoming assignment
- **Please keep your own code safely**