

# MP8 Overview Session

The background of the slide features a photograph of a large, classical-style statue of a woman in a long, flowing dress, standing with her arms outstretched. The statue is set in a park-like environment with many trees. The entire image is overlaid with a semi-transparent red color, which serves as a background for the white text.

**CS 240 - The University of Illinois**

Eunice Zhou

March 28, 2022

# Goals

- Create a versioned state server using *flask*
  - store key-value pairs of data with a version number
- Implement the API(s) that will allow users to interact with your web server
  - PUT, GET, DELETE request
- Explore two solutions to the same problem
  - store data locally in memory
  - store data remotely in a MongoDB database

# Versioned State Server

The background of the slide features a photograph of a large, classical-style statue of a woman, likely an allegorical figure, standing on a pedestal. The statue is surrounded by a crowd of people, some of whom are looking towards the camera. The entire image is overlaid with a semi-transparent orange color, which serves as a background for the white text.

# API Requirements

PUT /<key>

- Add a versioned object to storage
- Version number starts at 1
- Return `HTTP/200` if successful
- You can get the contents of the request as a UTF-8 string with `request.data.decode("utf-8")`
- Example: `PUT /date` with data content `"2022-03-26"`

# API Requirements

GET /<key>

- Retrieve the latest version of a key
- Return a JSON containing the latest value stored for <key> and the corresponding version number
- JSON format: { "value": "<string>", "version": <number> }
- Return [HTTP/200](#) if successful, [HTTP/404](#) if key not found
- Example JSON: { "value": "2022-03-28", "version": 3 }

# API Requirements

GET /<key>/<version>

- Retrieve a specific version of a key
- Similar to [GET /<key>](#) but for a specific version instead of the latest version
- Return [HTTP/200](#) if successful, [HTTP/404](#) if key not found
- Example: [GET /date/2](#)
- Example JSON: `{ "value": "2022-03-27", "version": 2 }`

# API Requirements

DELETE /<key>

- Completely delete all data associated with a key
- Delete the key and all version of the key
- Future versions of the key begin again with 1
- Return [HTTP/200](#) if successful

# Example

PUT /date

Content: "2022-03-26"

date

Version	Value
1	2022-03-26



# Example

PUT /date

Content: "2022-03-27"

date

Version	Value
1	2022-03-26
2	2022-03-27

# Example

GET /date

Result:

```
{ "value": "2022-03-27", "version": 2 }
```

date

Version	Value
1	2022-03-26
2	2022-03-27

# Example

GET /date/1

Result:

```
{ "value": "2022-03-26", "version": 1 }
```

date

Version	Value
1	2022-03-26
2	2022-03-27

# Example

GET /date/3

Result:

HTTP/404 Not Found

date

Version	Value
1	2022-03-26
2	2022-03-27

# Example

DELETE /date

Result:

HTTP/200 Success

date

Version	Value
1	2022-03-26
2	2022-03-27

# Part 1: Local Store

A photograph of a crowd gathered around a statue, overlaid with a semi-transparent orange filter. The statue is on a pedestal with the words "ALMA MATER" visible. The background shows bare trees.

# Local Storage

Complete `app.py` in `local-store/`

- Store the data locally using any Python data structure
- Launch the app with `python -m flask run`
- The server should be hosted on localhost (127.0.0.1) on port 5000

# Testing Part 1

Run your server and send HTTP requests using

- `curl` commands in the terminal
- applications such as Postman

To run the test suite

- `python -m pytest test_local.py`



# Part 2: MongoDB

A photograph of a crowd of people gathered around a statue of Alma Mater, overlaid with a semi-transparent orange filter. The text "Part 2: MongoDB" is centered in white. The background shows a large group of people, some looking towards the camera and others looking towards the statue. The statue is a large, classical-style figure, possibly representing a personification of wisdom or knowledge. The overall scene is outdoors, with trees visible in the background.

# MongoDB Data Store

Complete [app.py](#) in [mongodb-nosql/](#)

- Store the data in a MongoDB database
- You can run a MongoDB server using docker
  - `docker run --rm -it -p 27017:27017 mongo`
- Install [PyMongo](#) to connect to MongoDB in flask
  - Doc: <https://pymongo.readthedocs.io/en/stable/>
- The server should be hosted on `127.0.0.1:5001`

# Testing Part 2

Run your server and send HTTP requests using

- `curl` commands in the terminal
- applications such as Postman

(Optional) MongoDB Compass

To run the test suite

- `python -m pytest test_mongo.py`

# Question

The image features a semi-transparent orange overlay over a photograph of a crowd gathered around a statue. The statue is the central focus, with the words 'ALMA MATER' visible on its base. The crowd consists of many people, some looking towards the statue. The overall scene is captured in a warm, monochromatic orange tone.