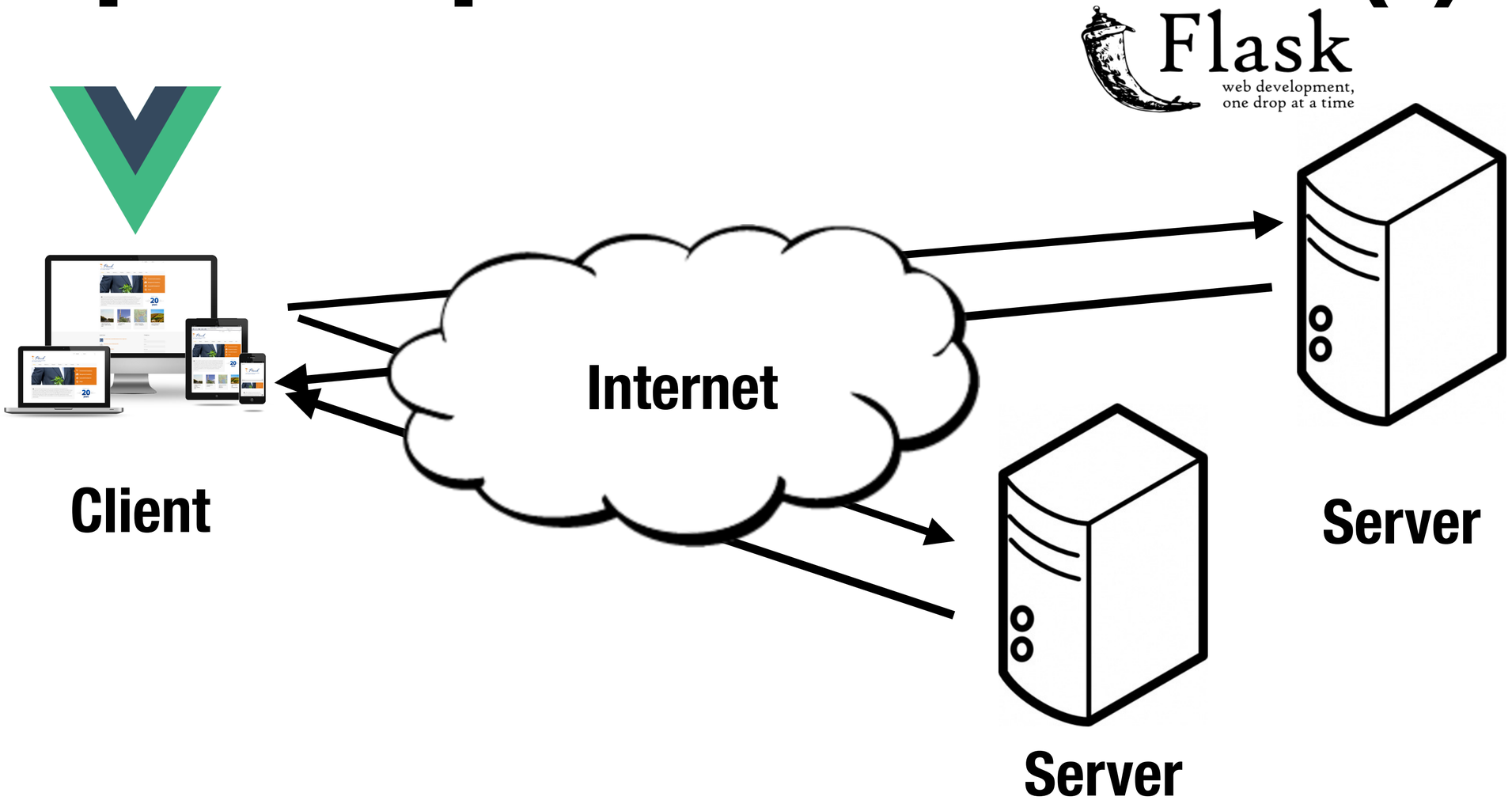
Web Programming
Decoupled REST API

Robert Ewald | University of Stavanger

Recap: Decoupled Client and server(s)



Webserver Role:

Flask application using templates:

- Couple data and presentation
- Adjust HTML documents
- Implements business logic
- Implement display logic
- Manipulate data with forms

Using Vue.js and AJAX

- Serve static HTML,JS,… files
- Serve data via AJAX & JSON
- Manipulate data via AJAX & JSON
- Can use data from other servers (if CORS allows)

Server-side APIs

- RESTful Web APIs
  - Accessing data independent from display
- Can maintain API independent from web application
- Can support different applications
- Can sell or offer the api to application developers

RESTful Web APIs

REST

- **RE**presentational **S**tate **T**ransfer
- REST is an architectural style (not a protocol)
  - Web service APIs are called RESTful
- Uniform interface separates clients from servers
  - Data storage is internal to the server
  - Servers are not concerned with the user's state
- Stateless
  - The client must provide all the information for the server to fulfill the request. No sessions.

Uniform interface

- Resources are identified by URIs
- Operations are performed on resources
- Resources are manipulated through representations
  - Representation contains enough information for the client to modify/ delete it on the server
  - Representations are typically in JSON or XML format

RESTful web APIs

- HTTP based
- Resources are identified by URIs
  - E.g., http://example.com/resources/
- Operations correspond to standard HTTP methods
  - GET, PUT, POST, DELETE
- Media type is JSON

Typical RESTful API

	GET	PUT	POST	DELETE
Collection URI `http://example.com/resources`	**List** elements	**Replace** the entire collection	**Create** a new element in the collection	**Delete** the entire collection
Element URI `http://example.com/resources/item17`	**Retrieve** the representation of an element	**Replace** element **create** if it doesn't exist	generally not used	**Delete** the element

Example

examples/ajax/vue/playlist

Example

examples/ajax/vue/playlist

```
@app.route("/playlist", methods=["GET"])
def getplace():
   ...

@app.route("/add", methods=["POST"])
def addSong():
    ...

@app.route("/remove", methods=["POST"])
def removeSong():
    ...
```

Not Rest.
If application grows, will be difficult to know what is removed.

Example

examples/ajax/vue/playlist-rest

```
@app.route("/playlist", methods=["GET"])
def getplace():
   ...

@app.route("/song", methods=["POST"])
def addSong():
    ...

@app.route("/song", methods=[“DELETE"])
def removeSong():
    ...
```

```
let response = await fetch("/song", {
    method: "DELETE",
    headers: {
        "Content-Type": "app
    },
    body: JSON.stringify({name: song.name, band: song.band}),
});
```

Can use **GET, POST, PUT, DELETE**

Exercises #1

github.com/dat310-2022/info/tree/main/
exercises/ajax/rest

References

- REST API tutorial
  - http://www.restapitutorial.com/