Intro to REST APIs

Adam Moore, GSB

What is REST?

Representational state transfer (REST) or RESTful web services are a way of providing interoperability between computer systems on the Internet. There are 6 constraints although most apis only care about 5.

Uniform Interface

However we implement our api, implement it in the same way for the entire api. There are some expectations.

Uniform Interface - Resources

Each endpoint is a resource. A resource is an object that is a representation of something. For example you don't get the database back from the request you get a representation of the database.

Uniform Interface - Identification of Resources

Use URIs

- /course
- /course/id/1234

Output Format Doesn't Matter

- JSON/XML

Don't Change The URI For Each Format

- No /course/xml, /course/json, etc.
- Use Accept header ex: Accept: application/json

Uniform Interface - Everything In The Response

Everything you need to modify or delete the resource is in the response.

Uniform Interface - Manipulate the Resources With the Same URI

If an endpoint is /course/id/1234. I can use the available verbs to manipulate that resource using the same uri.

Example

GET /course/id/1234 - returns details about the course with the id of 1234

DELETE /course/id/1234 - deletes the course with the id of 1234

Not

/course/id/1234/delete

Uniform Interface - Self-Descriptive Messages

The data returned also has enough information for the client to know what to do with it.

- Cache Headers
  - How long should I wait before asking for this data again?
- MIME Type
  - What format is this data in? JSON/XML?

Uniform Interface - Hypermedia as the Engine of Application State (HATEOAS)

Not always implemented, but can be very powerful.

```
{
    "cancel": "http://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708",
    "endDate": "2017-11-01T12:00:00-0700",
    "id": "http://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708",
    "name": "Architecture Council",
    "startDate": "2017-11-01T10:30:00-0700",
    "status": "Confirmed",
    "type": "BookingReservation"
}
```

Stateless

The server doesn't keep track of what happens from one request to the next.

Cacheable

Clients are allowed to cache responses. This means responses must implicitly or explicitly define themselves as able to be cached.

Client-Server

- Clients have no knowledge of how the server stores the data.
- Servers have no knowledge of how the client keeps track of its state.

Layered System

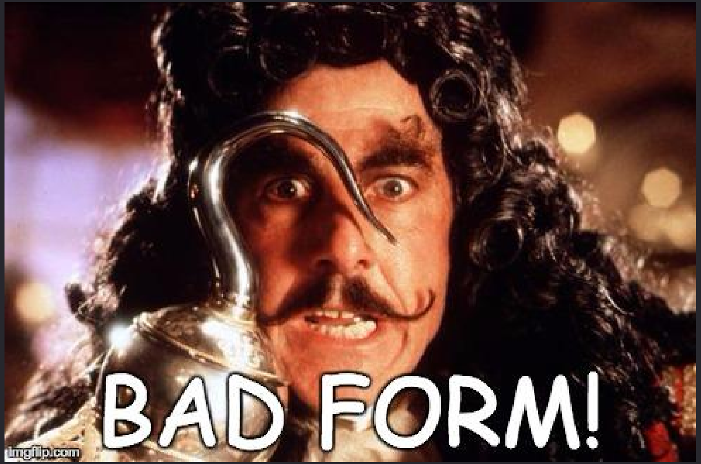
Systems may be put in place in front of the data server to improve cacheability, redundancy or provide other services like authorization.

Simple REST - The Internet

- GET
  - Return a webpage
- POST
  - Submit a form

Using GET for Form Submission

```
<form action="page.php" method="GET">
....
</form>
```



Verbs

- POST - Create new information
- GET - Read information
- DELETE - Delete the information
- PUT - Replace the information
- PATCH - Change the information

POST - https://api-dev.gsb.stanford.edu/booking/reservation

HTTP Status

201: Created - It also includes a Location header of the url to find what you just created

Body of POST

```
{
    "endDate": "2017-11-01T12:00:00-0700",
    "name": "Architecture Council",
    "startDate": "2017-11-01T10:30:00-0700",
    "type": "BookingReservation"
}
```

GET - https://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708

HTTP Status

200: OK - With etag in the header

Return Body

```
{
   "cancel": "http://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708",
   "endDate": "2017-11-01T12:00:00-0700",
   "id": "http://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708",
   "name": "Architecture Council",
   "startDate": "2017-11-01T10:30:00-0700",
   "status": "Confirmed",
   "type": "BookingReservation"
}
```

DELETE - https://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708

HTTP Status

204: No Content

PUT - https://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708

HTTP Status

204: No Content

Body of POST

```
{
    "endDate": "2017-11-01T12:00:00-0700",
    "name": "Architecture Council",
    "startDate": "2017-11-01T10:30:00-0700",
    "type": "BookingReservation"
}
```

PATCH - https://api-dev.gsb.stanford.edu/booking/reservation/id/ems-728708

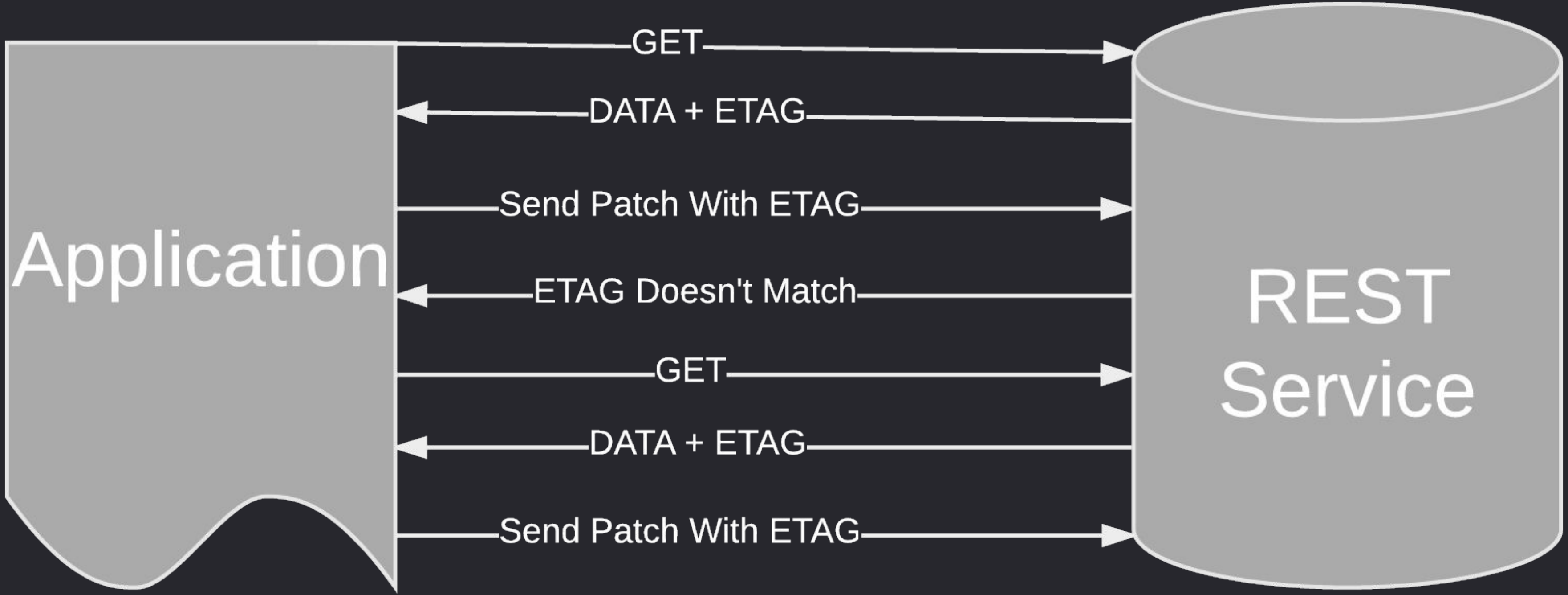
Status Code

204: No Content

PATCH Body

```
[
  { "op": "replace", "path": "/name", "value": "My Reservation" },
  { "op": "add", "path": "/description", "value": ["Here is my description"] },
  { "op": "remove", "path": "/phone"}
]
```

PATCH - Process


Application
REST Service
GET
DATA + ETAG
Send Patch With ETAG
ETAG Doesn't Match
GET
DATA + ETAG
Send Patch With ETAG