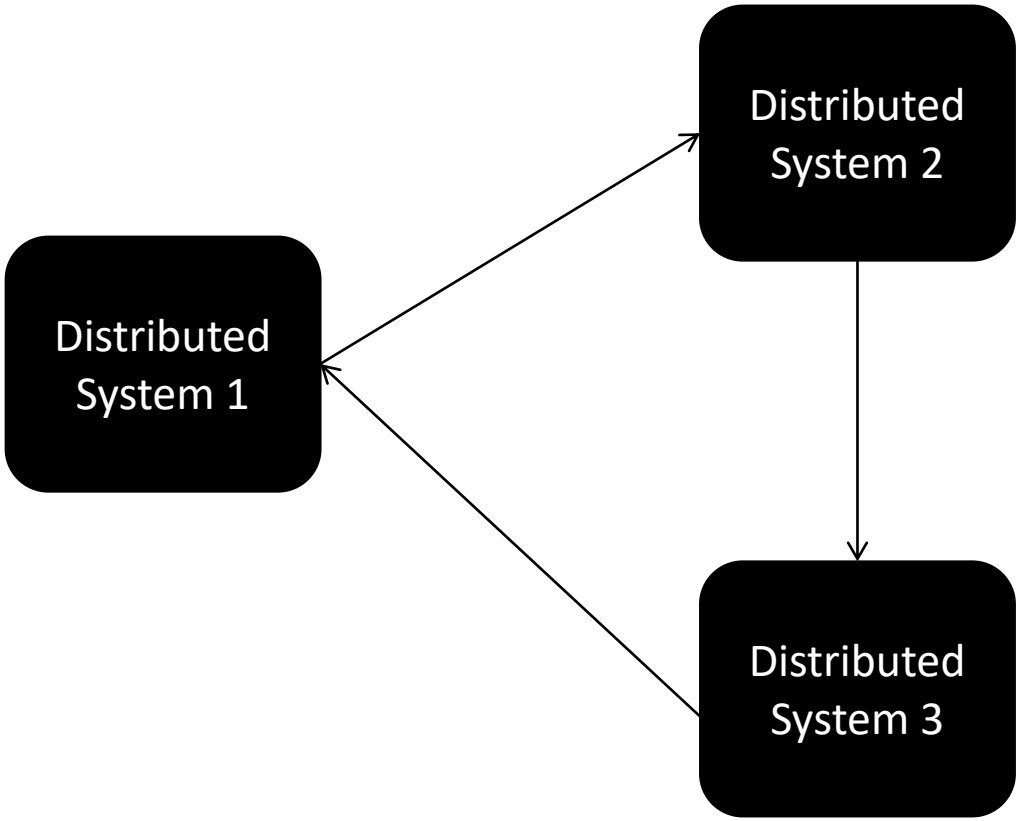
Web Application Development and Web Services

Venkatesh Vinayakarao
venkateshv@cmi.ac.in
http://vvtesh.co.in

Chennai Mathematical Institute

If You Think Math is Hard Try Web Design. **– PixxelzNet.**

How to Achieve Interoperability?

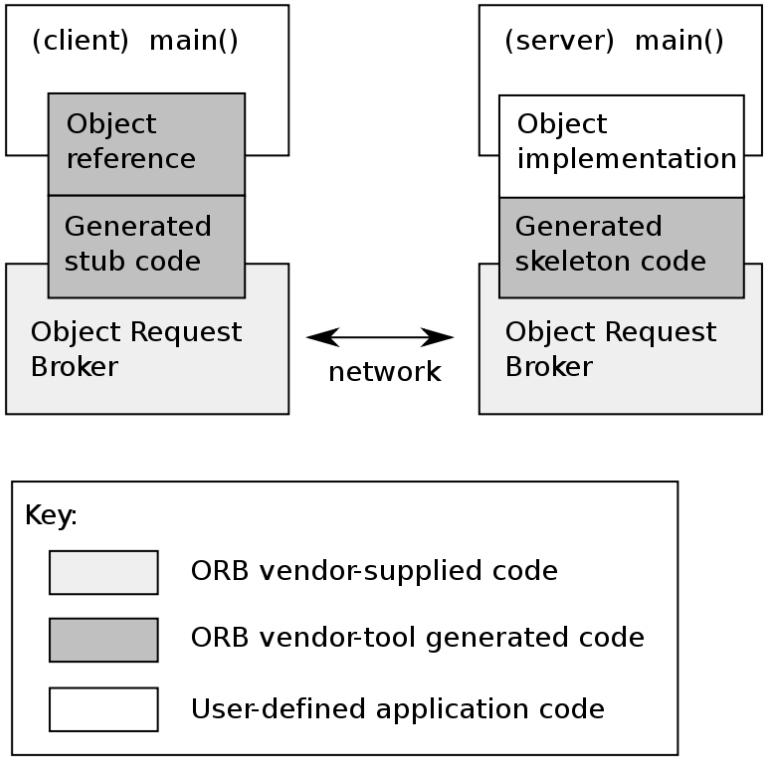


Interoperability Solutions

- Many Solutions
  - File Transfer
  - Shared DB
  - Remote Procedure Calls
  - Message Passing
- Middleware platforms aimed at making it more structured and easier
  - CORBA, DCOM, RMI, …
  - Web Services

Interoperability Solutions

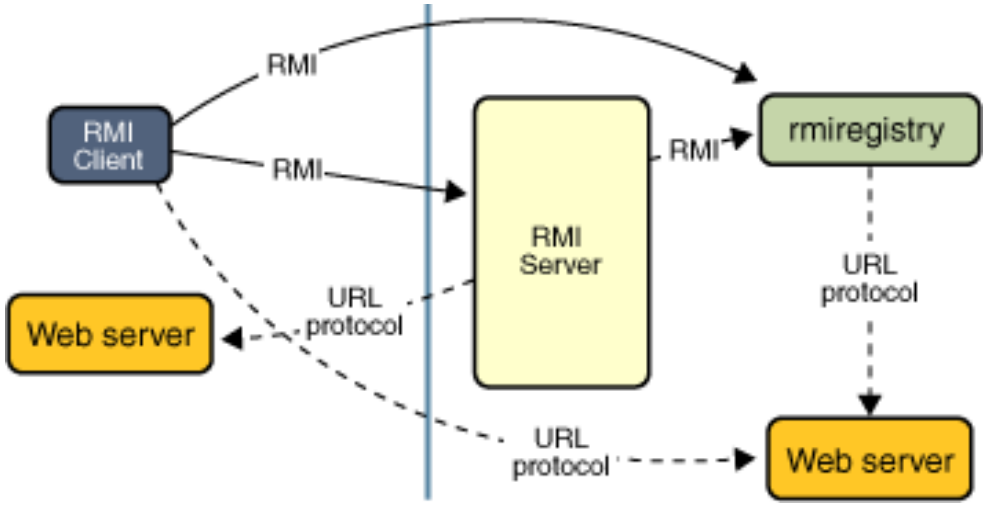
- CORBA (1991)
  - Standards-based, vendor-neutral, and language-agnostic.
  - Communicate by message passing over network
  - Read Corba: Gone But (Hopefully) Not Forgotten, Queue Vol 5, No. 4.



https://www.omg.org/spec/CORBA/
https://en.wikipedia.org/wiki/Common_Object_Request_Broker_Architecture
https://docs.oracle.com/javase/8/docs/technotes/guides/idl/jidlExample.html

More Interoperability Solutions

- Distributed Component Object Model (DCOM) (Microsoft)
- RMI (Sun Microsystems)
  - Invoke method on a remote object.



https://docs.oracle.com/javase/tutorial/rmi/overview.html

Web Services

- A “service” is a software component provided through an (often, network-accessible) endpoint.
- Service consumer and provider use messages to exchange invocation request and response information in the form of self-containing documents.

What do you understand by “**Web**”?

Early Static Web

- Developed in 1990 at CERN
- NCSA Mosaic 1.0 was the first browser, released by the National Center for Supercomputer Applications (NCSA).

The Dynamic Web

- Httpd 1.0 web server allowed Common Gateway Interface (CGI).
- CGI allows a browser client to request data from a program running on a Web server.

CGI Script

```
#! /usr/local/bin/perl
# Display the form data set sent in a GET or POST request.

print "Content-type: text/html\n\n";
print "<html><head><title>Form Data</title></head> \n";
print "<body bgcolor=\"#FFFFFF\"\n>"

if ($ENV{'REQUEST_METHOD'} eq 'POST') {
    read (STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
    @pairs = split(/&/, $buffer);
} elsif ($ENV{'REQUEST_METHOD'} eq 'GET') {
    @pairs = split(/&/, $ENV{'QUERY_STRING'});
} else {
    print "<p>$ENV{'REQUEST_METHOD'} message received</p>";
}
foreach $pair (@pairs) {
    ($name, $value) = split(/=/, $pair);
    $value =~ tr/+/ /;
    $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $name =~ tr/+/ /;
    $name =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    print "<p>Field $name has the value $value</p> \n";
    $FORM{$name} = $value;
} print "</body></html> \n";
```

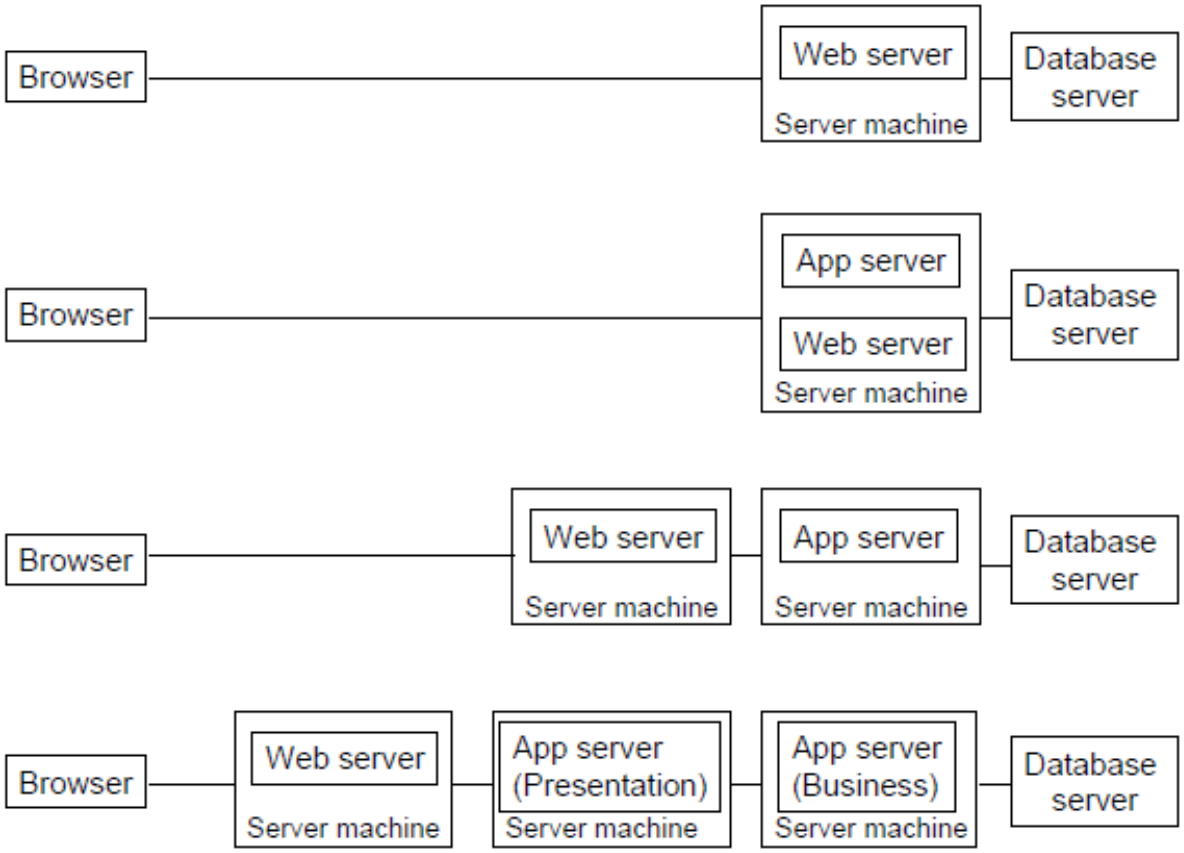
Server-Side (javascript) Scripting

```
<html>
    <head>  <title>Server-Side JavaScript Example Author Listing</title>  </head>
    <body>
        <h1>Author List</h1>
        <server>
            if (!database.connected()){
                database.connect("ODBC","bookdb","admin","","");
            }
            if (database.connected()) {
                qs = "SELECT au_id, au_fname, au_lname FROM authors";
                results =  database.cursor(qs);
                write("<table border=2 cellpadding=2 cellspacing=2>" +
                      "<tr><th>ID</th><th>First Name</th><th>Last Name </th></tr> \n");
                while(results.next()) {
                      write("<tr><td>" + results.au_id + "</td> + "<td>" +
                            results.au_fname + "</td>" +
                            "<td>" + results.au_lname + "</td></tr> \n");
                }
                results.close(); write("</table> \n");
            }
            else {
                write("<p>Database connection failed");
            }
        </server>
    </body>
</html>
```

ASP Page

```
<%
    Dim conn, rs
    Set conn = Server.CreateObject("ADODB.Connection")
    Set rs = Server.CreateObject("ADODB.Recordset")
    conn.Open "bookdb", "sa", "password"
    Set rs = conn.Execute("select au_id, au_fname, au_lname from authors")
%>
<html>
    <head>  <title>ASP Example Author Listing</title></head>
    <body>
        <h1>Author List</h1>
        <table>
            <tr><th>ID</th><th>First Name</th><th>Last Name</th></tr>
            < % Do Until rs.EOF %>
                <tr><td><%=rs("au_id") %></td>
                <td><%=rs("au_fname") %></td>
                <td><%=rs("au_lname") %></td></tr>
                < % rs.movenext
                Loop
            %>
        < /table>
    < /body>
< /html>
```

Evolution of Web and App Servers



Software as a Service (SaaS)

https://od-api.oxforddictionaries.com/api/v2/entries/en-us/ubiquitous

```
7   # TODO: replace with your own app_id and app_key
8   app_id = '<my app_id>'
9   app_key = '<my app_key>'
10  language = 'en'
11  word_id = 'Ace'
12
13  url = 'https://od-api.oxforddictionaries.com:443/api/v2/entries/'  + language + '/'  + word_id.lower()
14  #url Normalized frequency
15  urlFR = 'https://od-api.oxforddictionaries.com:443/api/v2/stats/frequency/word/'  + language + '/?corpus=nmc&lemma='
+ word_id.lower()
16
17  r = requests.get(url, headers = {'app_id' : app_id, 'app_key' : app_key})
18
19  print("code {}\n".format(r.status_code))
20  print("text \n" + r.text)
21  print("json \n" + json.dumps(r.json()))
```

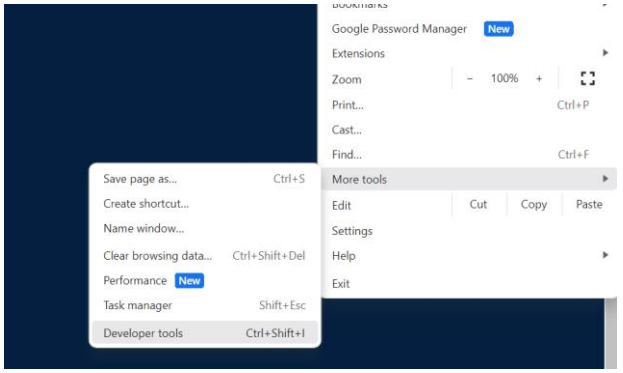
API Service from Oxford Dictionary
https://developer.oxforddictionaries.com/

{
"definitions": [
"present, appearing, or found everywhere"]
}

Response in JSON format

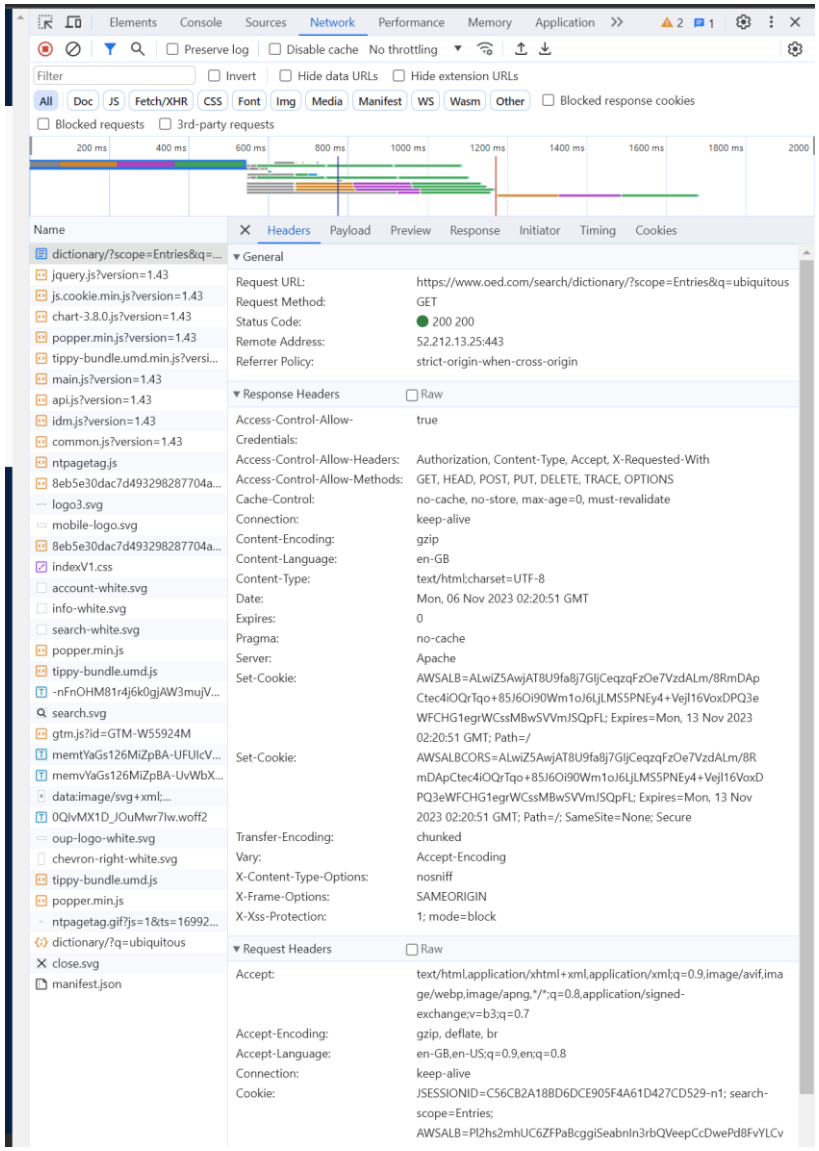
Try this!

- Visit https://www.oed.com/
- Type some word in the dictionary search bar.
- Go to browser settings -> More tools -> Developer tools.
- Open Network tab.
- Hit the search button on the search bar.



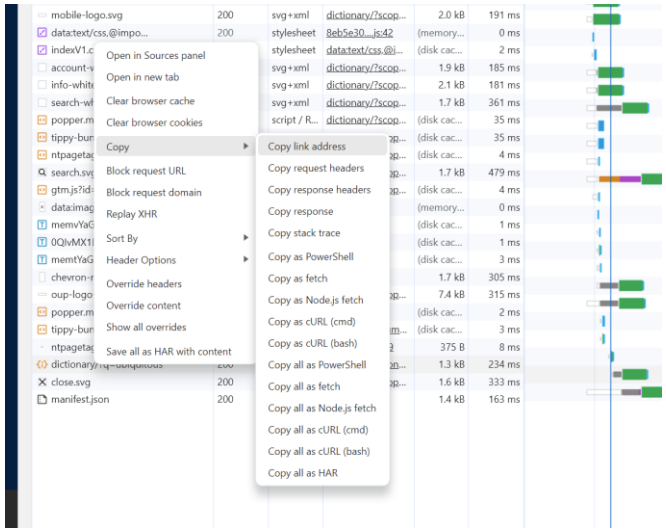
Try this!

- Click on the following entry in the network tab - https://www.oed.com/search/dictionary/?scope=Entries&q=ubiquitous
- You will see the individual requests made by the site and their response details.



Try this!

- Copy this link address and run it in a new browser tab - https://www.oed.com/autocomplete/dictionary/?q=ubiquitous

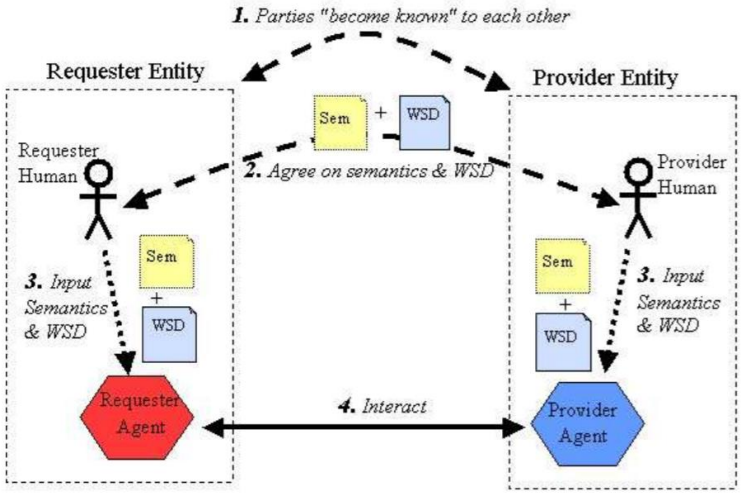


Response

[{"name":"ubiquitous","count":null,"label":"ubiquitous","path":null},{"name":"ubiquitously","count":null,"label":"ubiquitously","path":null},{"name":"ubiquitousness","count":null,"label":"ubiquitousness","path":null},{"name":"ubiquitous computing","count":null,"label":"ubiquitous computing","path":null}]

Web Services

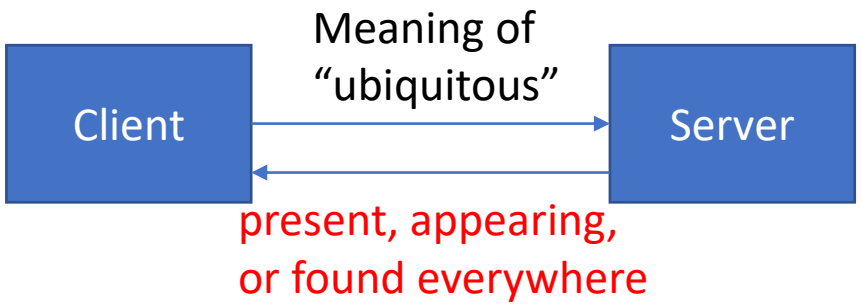
- A Web service is a software system designed to support interoperable machine-to-machine interaction over a network.

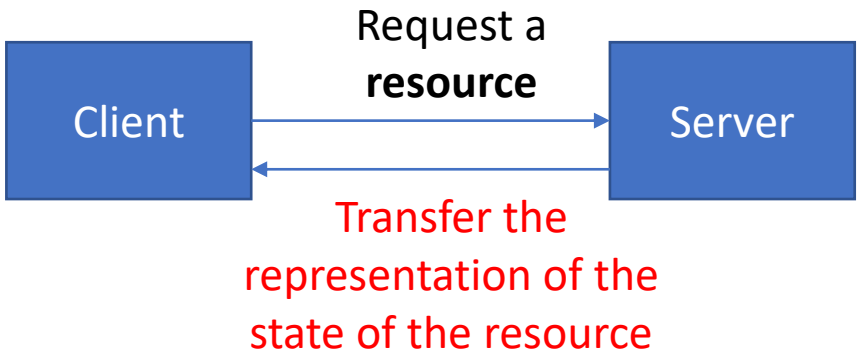


https://www.w3.org/TR/ws-arch/wsa.pdf

REST API

- REST = Representational State Transfer
  - Proposed by Roy Fielding in 2000.





Resource

- Any information that can be named is a resource
  - Document, image, or any other object.
- Description of the state of the resource at any timestamp is known as resource representation
  - Representation consists of data describing the resource.
- Resource methods are used to transfer the resource state representations.
  - Need not be always HTTP (GET/POST/…).

RESTful Web Services API

- Let us retrieve an existing configuration:
  - http://example.com/network-app/configurations/678678
  - HTTP GET /configurations/{id}

- Similarly, we can POST, PUT, and DELETE.
  - HTTP POST /devices
  - HTTP POST /configurations
  - HTTP PUT /devices/{id}/configurations
  - HTTP DELETE /devices/{id}/configurations/{id}

https://restfulapi.net/rest-api-design-tutorial-with-example/

HTTP

- HTTP Methods

HTTP Method	Purpose
POST	Create
GET	Retrieve
PUT	Update
DELETE	Delete

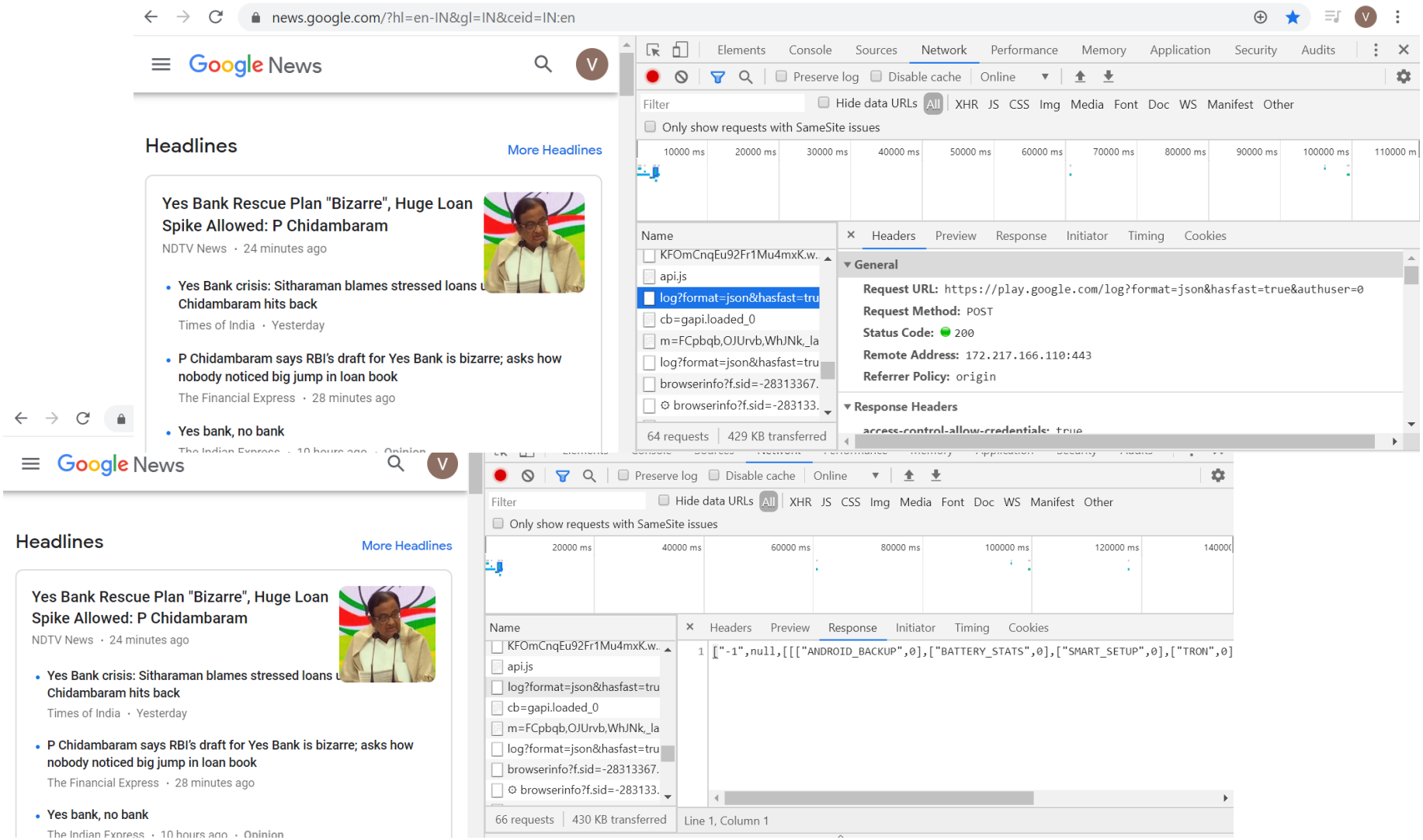
- *"An idempotent HTTP method is an HTTP method that can be called many times without different outcomes."*
  - POST is NOT idempotent.
  - GET, PUT, DELETE are idempotent.

HTTP Response Codes

- 2xx
  - Success
  - Example: 200 = OK, 201 = Created, 202 = Accepted (if it is a long-running task)
- 4xx
  - Client Error
  - Example: 400 = Bad Request, 404 = Not Found.
- 5xx
  - Server Error
  - Example: 500 = Internal Server Error

https://restfulapi.net/http-status-codes/

REST in Real World



Designing REST API

- Identify the object model
- Create Model URIs
- Determine Representations
- Assign HTTP Methods

```
<device id="12345">
    <link rel="self" href="/devices/12345"/>
    <deviceFamily>apple-es</deviceFamily>
    <OSVersion>10.3R2.11</OSVersion>
    <platform>SRX100B</platform>
    <serialNumber>32423457</serialNumber>
    <connectionStatus>up</connectionStatus>
    <ipAddr>192.168.21.9</ipAddr>
    <name>apple-srx_200</name>
    <status>active</status>
</device>
```

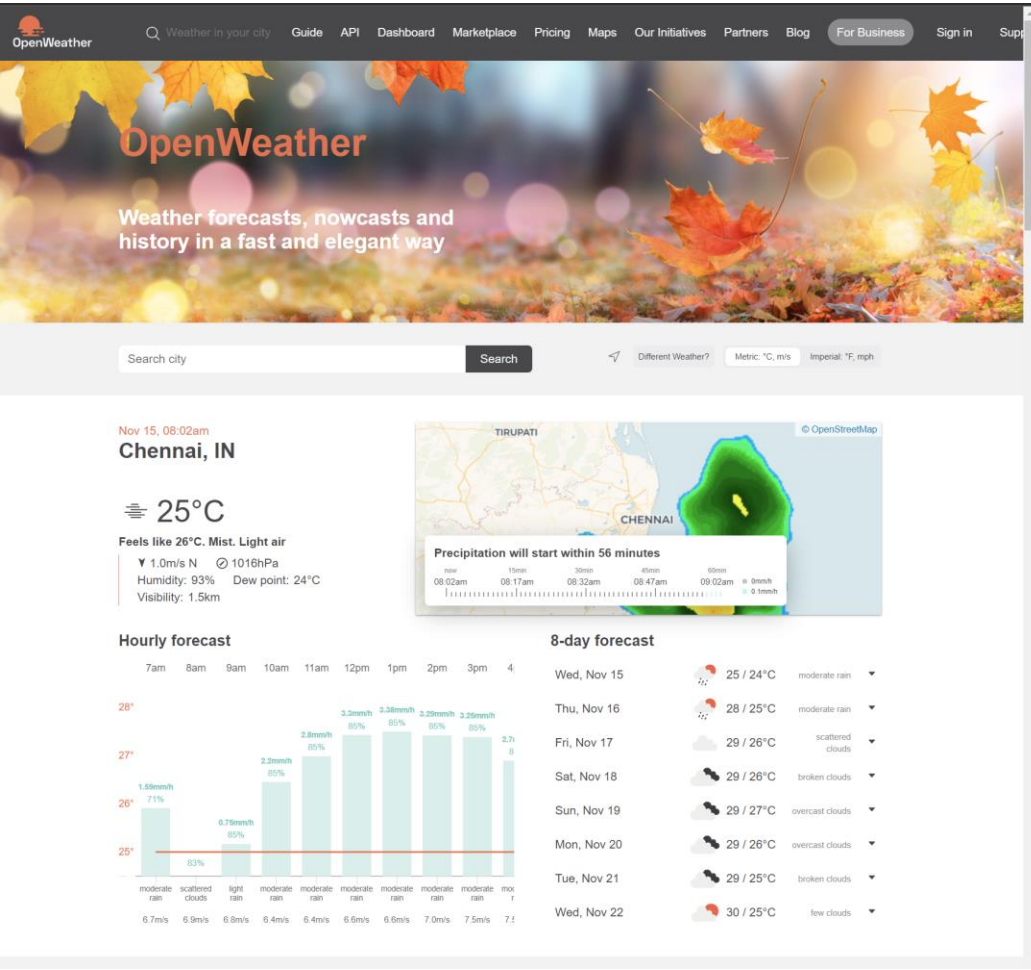
Web Services for a Banking Application

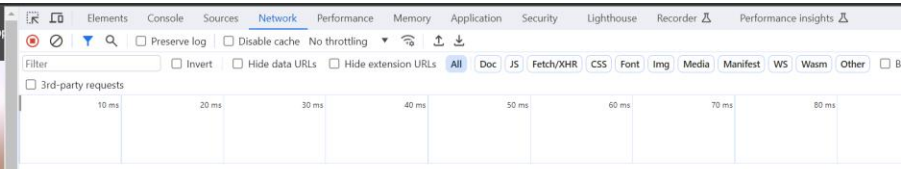
- Designing the REST API
  - Object Model
    - Customer, Account
  - Create Model URIs
    - /customers/{customerId}
    - /customers/{customerId}/accounts
    - /customers/{customerId}/accounts/{accountId}
  - Determine Representations
    - Represent all Account information as an XML/JSON
    - Represent all Customer information as XML/JSON
  - Assign HTTP Methods
    - Open Account = Create an Account Resource ➔ HTTP POST
    - Close Account = Delete the Account ➔ HTTP DELETE

Weather Example

- See https://openweathermap.org

https://api.openweathermap.org/data/2.5/weather?lat={lat}&lon={lon}&appid={API key}

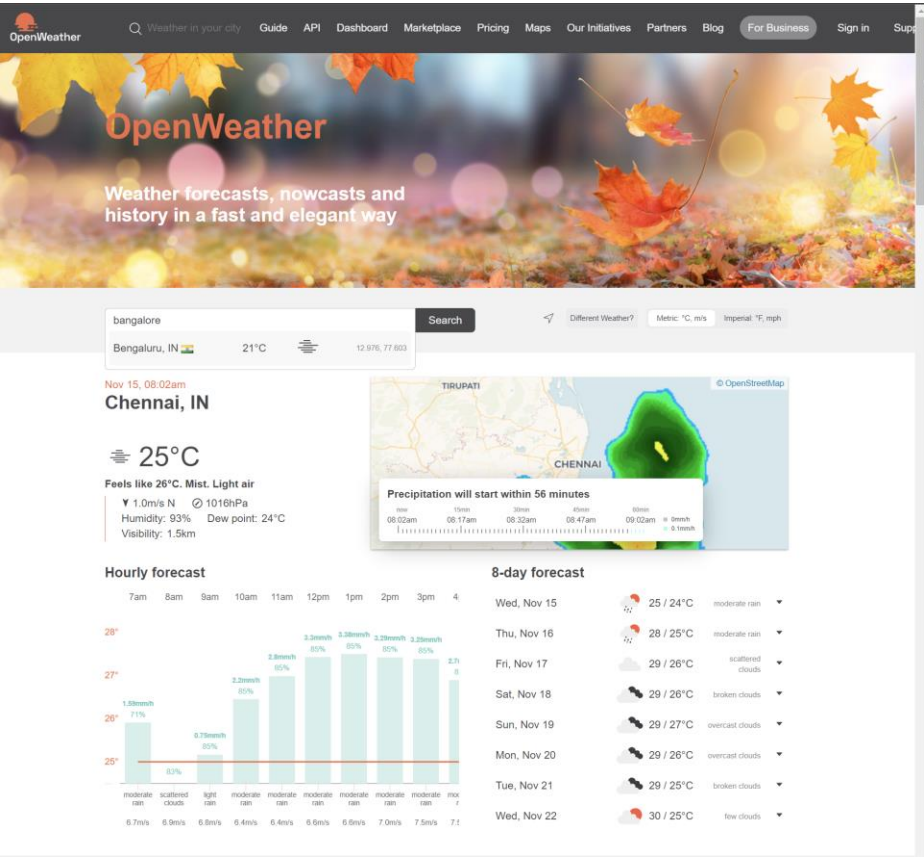


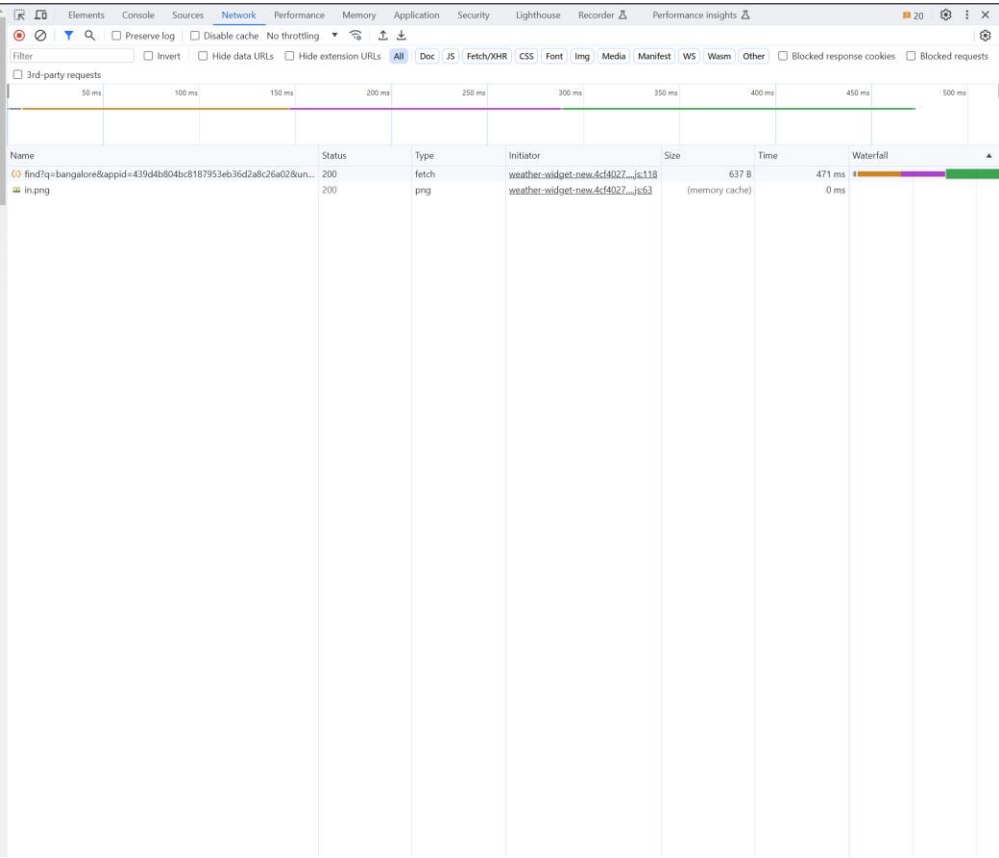


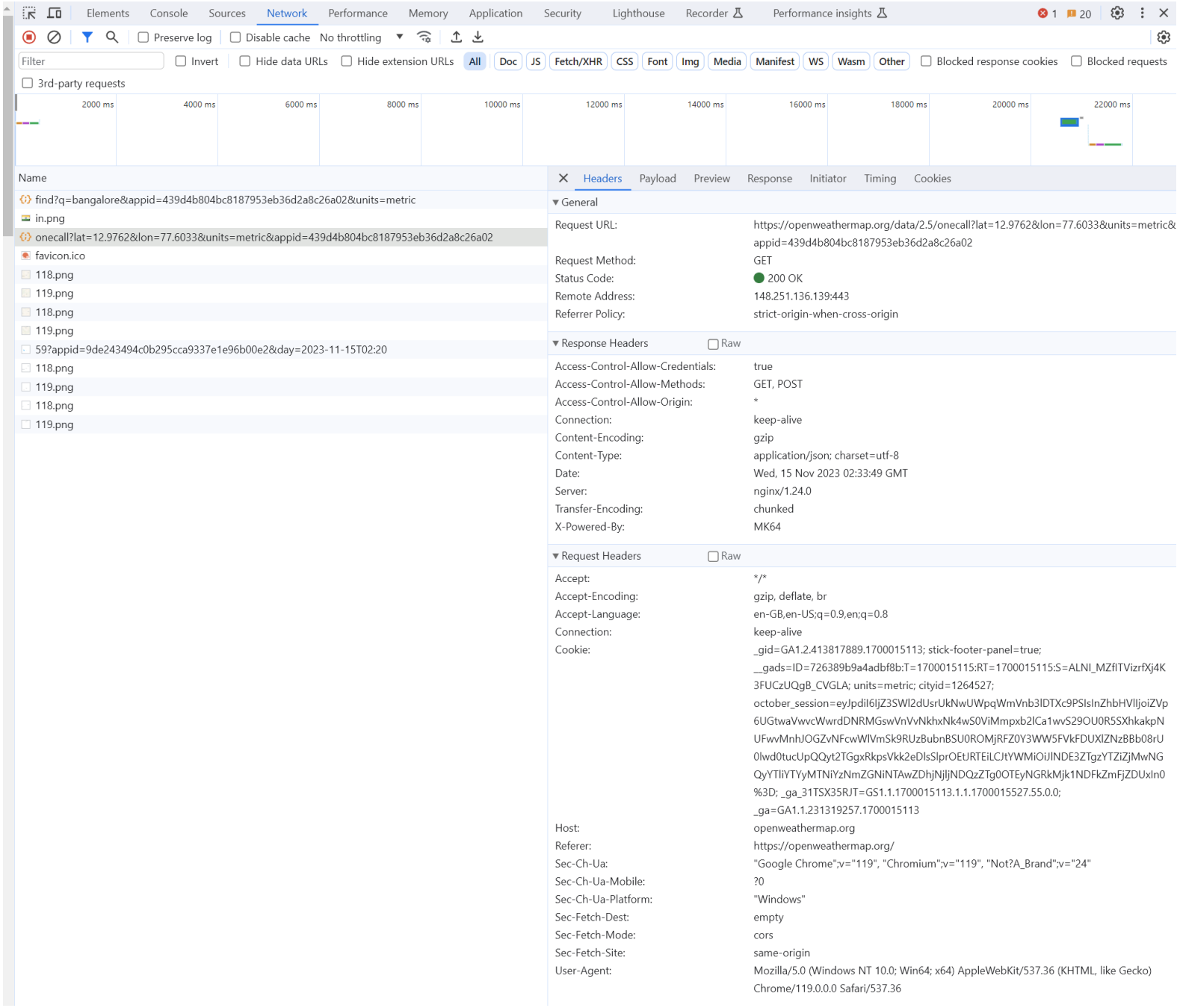
Recording network activity...

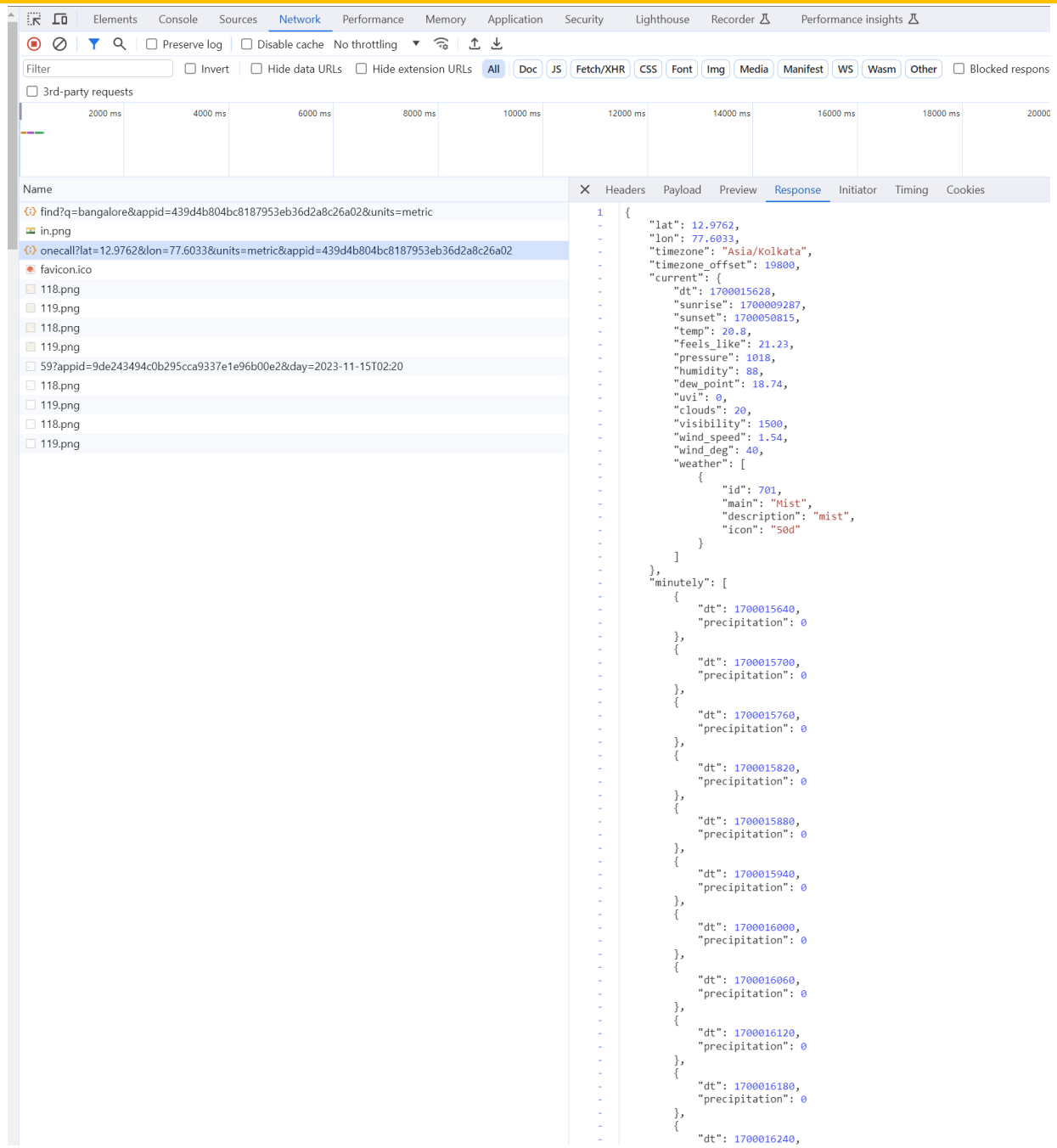
Perform a request or hit **Ctrl + R** to record the reload.

Learn more


OpenWeather
Weather in your city
Guide
API
Dashboard
Marketplace
Pricing
Maps
Our Initiatives
Partners
Blog
For Business
Sign in
OpenWeather
Weather forecasts, nowcasts and history in a fast and elegant way
bangalore
Search
Bengaluru, IN
21°C
Different Weather?
Metric: °C, m/s
Imperial: °F, mph
Nov 15, 08:02am
Chennai, IN
25°C
Feels like 26°C. Mist. Light air
1.0m/s N
1016hPa
Humidity: 93%
Dew point: 24°C
Visibility: 1.5km
TIRUPATI
CHENNAI
© OpenStreetMap
Precipitation will start within 56 minutes
08:02am
08:17am
08:32am
08:47am
09:02am
Hourly forecast
7am
8am
9am
10am
11am
12pm
1pm
2pm
3pm
8-day forecast
Wed, Nov 15
25 / 24°C
moderate rain
Thu, Nov 16
28 / 25°C
moderate rain
Fri, Nov 17
29 / 26°C
scattered clouds
Sat, Nov 18
29 / 26°C
broken clouds
Sun, Nov 19
29 / 27°C
overcast clouds
Mon, Nov 20
29 / 26°C
overcast clouds
Tue, Nov 21
29 / 25°C
broken clouds
Wed, Nov 22
30 / 25°C
few clouds


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Performance insights
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Doc
JS
Fetch/XHR
CSS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
Name
Status
Type
Initiator
Size
Time
Waterfall
find?q=bangalore&appid=439d4b804bc8187953eb36d2a8c26a02&un...
200
fetch
weather-widget-new.4cf4027...js:118
637 B
471 ms
in.png
200
png
weather-widget-new.4cf4027...js:63
(memory cache)
0 ms


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Performance insights
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Doc
JS
Fetch/XHR
CSS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
2000 ms
4000 ms
6000 ms
8000 ms
10000 ms
12000 ms
14000 ms
16000 ms
18000 ms
20000 ms
22000 ms
Name
find?q=bangalore&appid=439d4b804bc8187953eb36d2a8c26a02&units=metric
in.png
onecall?lat=12.9762&lon=77.6033&units=metric&appid=439d4b804bc8187953eb36d2a8c26a02
favicon.ico
118.png
119.png
118.png
119.png
59?appid=9de243494c0b295cca9337e1e96b00e2&day=2023-11-15T02:20
118.png
119.png
118.png
119.png
Headers
Payload
Preview
Response
Initiator
Timing
Cookies
General
Request URL:
https://openweathermap.org/data/2.5/onecall?lat=12.9762&lon=77.6033&units=metric&appid=439d4b804bc8187953eb36d2a8c26a02
Request Method:
GET
Status Code:
200 OK
Remote Address:
148.251.136.139:443
Referrer Policy:
strict-origin-when-cross-origin
Response Headers
Raw
Access-Control-Allow-Credentials:
true
Access-Control-Allow-Methods:
GET, POST
Access-Control-Allow-Origin:
*
Connection:
keep-alive
Content-Encoding:
gzip
Content-Type:
application/json; charset=utf-8
Date:
Wed, 15 Nov 2023 02:33:49 GMT
Server:
nginx/1.24.0
Transfer-Encoding:
chunked
X-Powered-By:
MK64
Request Headers
Raw
Accept:
/
Accept-Encoding:
gzip, deflate, br
Accept-Language:
en-GB,en-US;q=0.9,en;q=0.8
Connection:
keep-alive
Cookie:
_gid=GA1.2.413817889.1700015113; stick-footer-panel=true;
__gads=ID=726389b9a4adbf8b:T=1700015115:RT=1700015115:S=ALNI_MZfITVizrfXj4K3FUCzUQgB_CVGLA; units=metric; cityid=1264527;
october_session=eyJpdiI6IjZ3SWl2dUsrUkNwUWpqWmVnb3lDTXc9PSIsInZhbHVlIjoiZVp6UGtwaVwvcWwrdDNRMGswVnVvNkhxNk4wS0ViMmpxb2lCa1wvS29OU0R5SXhkakpNUFwvMnhJOGZvNFcwWlVmSk9RUzBubnBSU0ROMjRFZ0Y3WW5FVkFDUXlZNzBBb08rU0lwd0tucUpQQyt2TGgxRkpsVkk2eDlsSlprOEtJRTEiLCJtYWMiOiJlNDE3ZTgzYTZiZjMwNGQyYTliYTYyMTNiYzNmZGNiNTAwZDhjNjljNDQzZTg0OTEyNGRkMjk1NDFkZmFjZDUxIn0%3D; _ga_31TSX35RJT=GS1.1.1700015113.1.1.1700015527.55.0.0;
_ga=GA1.1.231319257.1700015113
Host:
openweathermap.org
Referer:
https://openweathermap.org/
Sec-Ch-Ua:
"Google Chrome";v="119", "Chromium";v="119", "Not?A_Brand";v="24"
Sec-Ch-Ua-Mobile:
?0
Sec-Ch-Ua-Platform:
"Windows"
Sec-Fetch-Dest:
empty
Sec-Fetch-Mode:
cors
Sec-Fetch-Site:
same-origin
User-Agent:
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Performance insights
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Doc
JS
Fetch/XHR
CSS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked respons
3rd-party requests
2000 ms
4000 ms
6000 ms
8000 ms
10000 ms
12000 ms
14000 ms
16000 ms
18000 ms
20000
Name
find?q=bangalore&appid=439d4b804bc8187953eb36d2a8c26a02&units=metric
in.png
onecall?lat=12.9762&lon=77.6033&units=metric&appid=439d4b804bc8187953eb36d2a8c26a02
favicon.ico
118.png
119.png
118.png
119.png
59?appid=9de243494c0b295cca9337e1e96b00e2&day=2023-11-15T02:20
118.png
119.png
118.png
119.png
Headers
Payload
Preview
Response
Initiator
Timing
Cookies
{
"lat": 12.9762,
"lon": 77.6033,
"timezone": "Asia/Kolkata",
"timezone_offset": 19800,
"current": {
"dt": 1700015628,
"sunrise": 1700009287,
"sunset": 1700050815,
"temp": 20.8,
"feels_like": 21.23,
"pressure": 1018,
"humidity": 88,
"dew_point": 18.74,
"uvi": 0,
"clouds": 20,
"visibility": 1500,
"wind_speed": 1.54,
"wind_deg": 40,
"weather": [
{
"id": 701,
"main": "Mist",
"description": "mist",
"icon": "50d"
}
]
},
"minutely": [
{
"dt": 1700015640,
"precipitation": 0
},
{
"dt": 1700015700,
"precipitation": 0
},
{
"dt": 1700015760,
"precipitation": 0
},
{
"dt": 1700015820,
"precipitation": 0
},
{
"dt": 1700015880,
"precipitation": 0
},
{
"dt": 1700015940,
"precipitation": 0
},
{
"dt": 1700016000,
"precipitation": 0
},
{
"dt": 1700016060,
"precipitation": 0
},
{
"dt": 1700016120,
"precipitation": 0
},
{
"dt": 1700016180,
"precipitation": 0
},
{
"dt": 1700016240,

Implementing RESTful web services

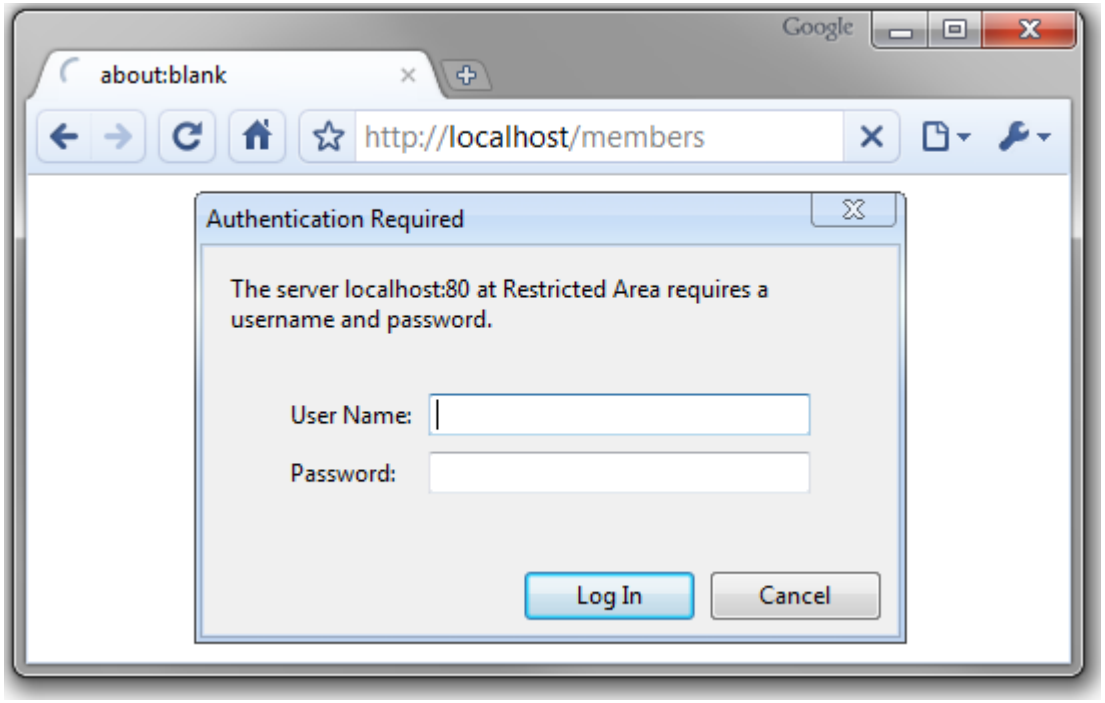
- Java API for RESTful web services (JAX-RS) [JSR 311] is specification.
- Jersey is a popular JAX-RS implementation.
- JAX-RS Annotations helps in building web services

```
@Path("/configurations")
public class ConfigurationResource
{
    @Path("/{id}")
    @GET
    public Response getConfigurationById(@PathParam("id") Integer id){
        ...
    }
}
```

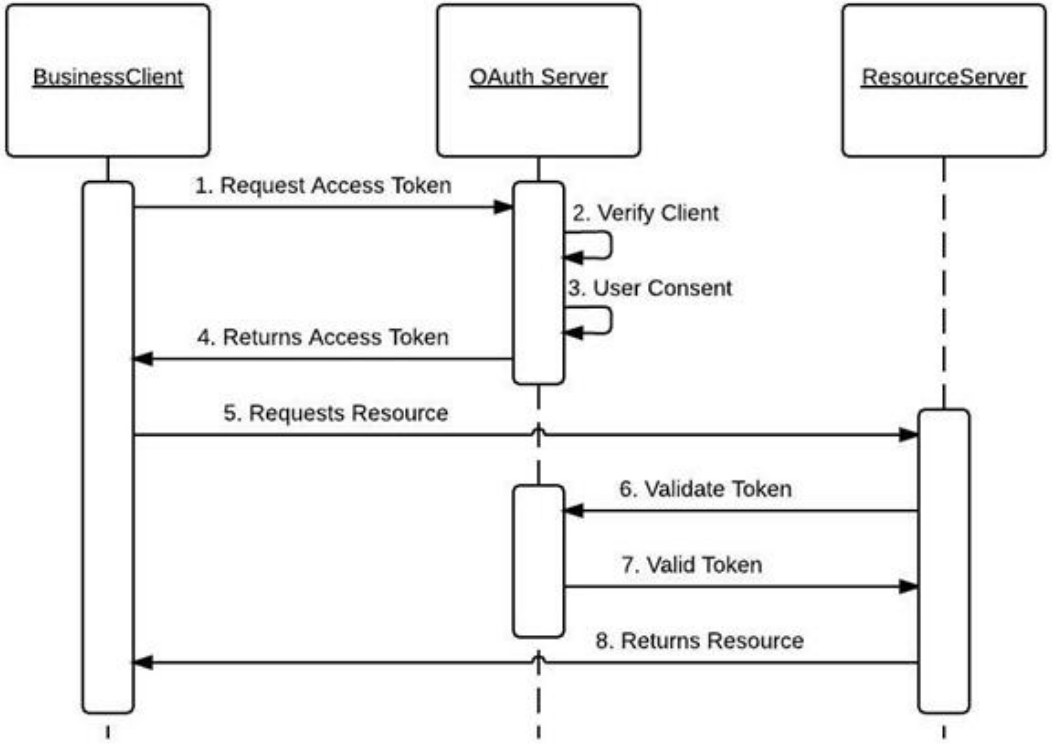
Authentication

- Basic HTTP Authentication
  - User enters the credentials
- Query String Authentication
  - URL has the credentials
- API Keys
  - Sever generated keys are used to identify the user.
- Token-based Authentication
  - oAuth method
  - Most secure form of authentication out of these four.

Basic HTTP Authentication



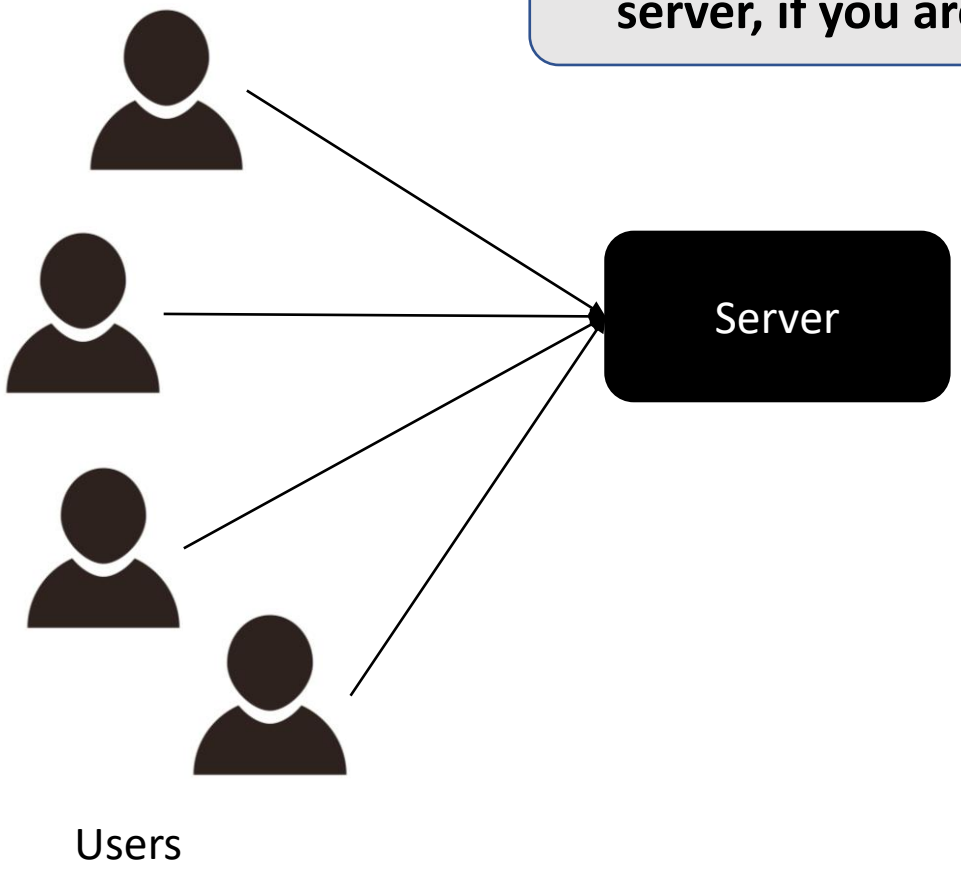
oAuth 2.0 Architecture



https://docs.oracle.com/cd/E82085_01/160027/JOS%20Implementation%20Guide/Output/oauth.htm

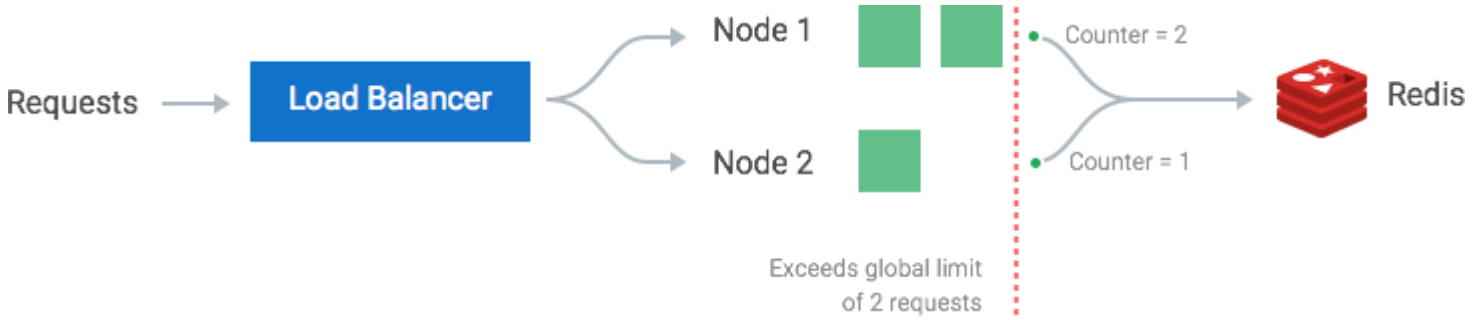
Web Services – Rate Limiting

Can you think of a way to bring down a server, if you are one of the users?



Rate Limiting

- A Leaky Bucket Solution
  - Queue up and service at a specific rate.
- Fixed Window Approach
  - Every request is served in a fixed time slot.
  - If the counter exceeds a threshold, the request is discarded.



https://konghq.com/blog/how-to-design-a-scalable-rate-limiting-algorithm/

Scaling

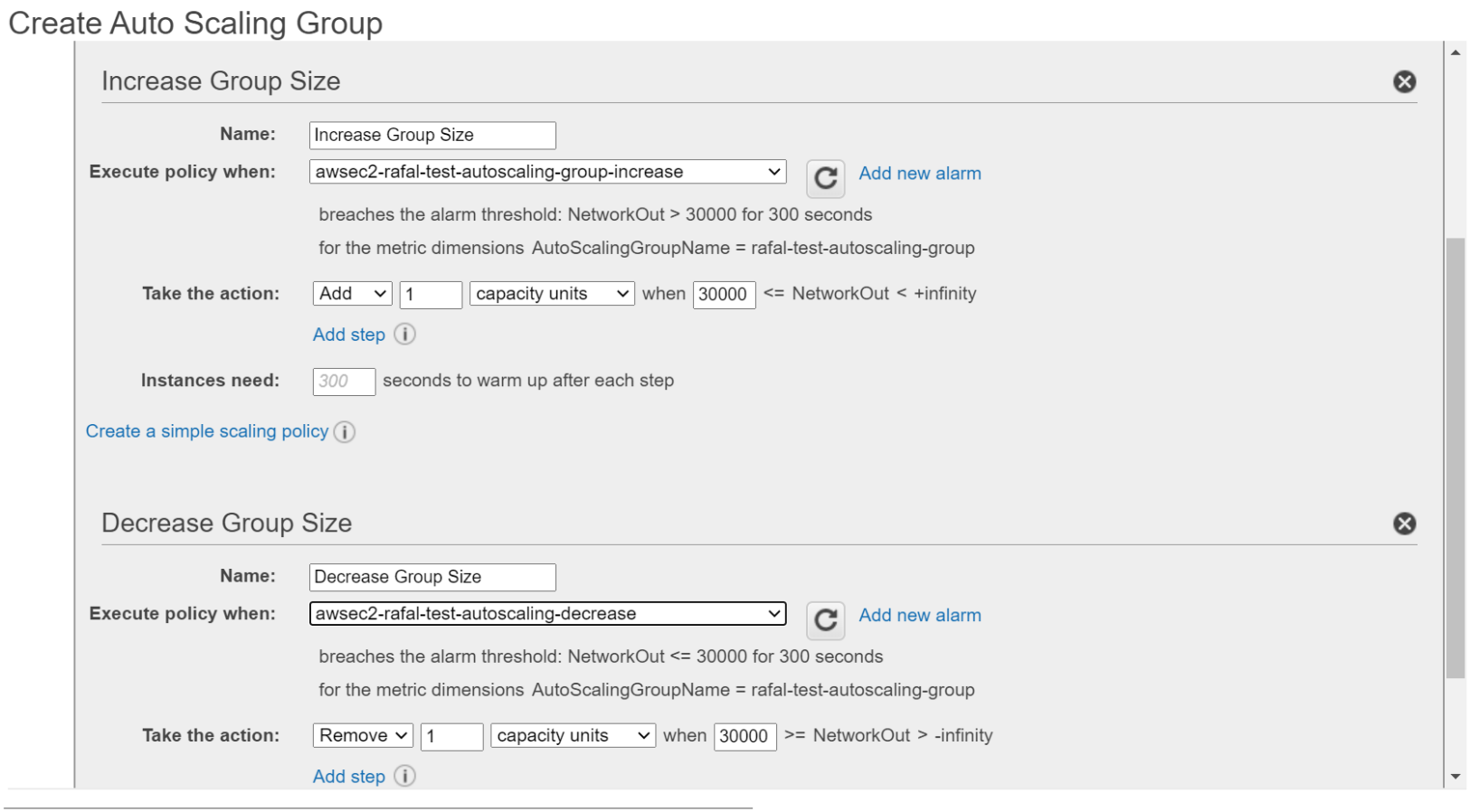
- Need for scaling
  - Traffic volume is different at different times
  - Keeping lesser number of server instances saves cost
  - Increase the server instances only when traffic is high
- Manual Scaling
  - May not work in situations where we cannot predict traffic patterns

Server Auto-Scaling

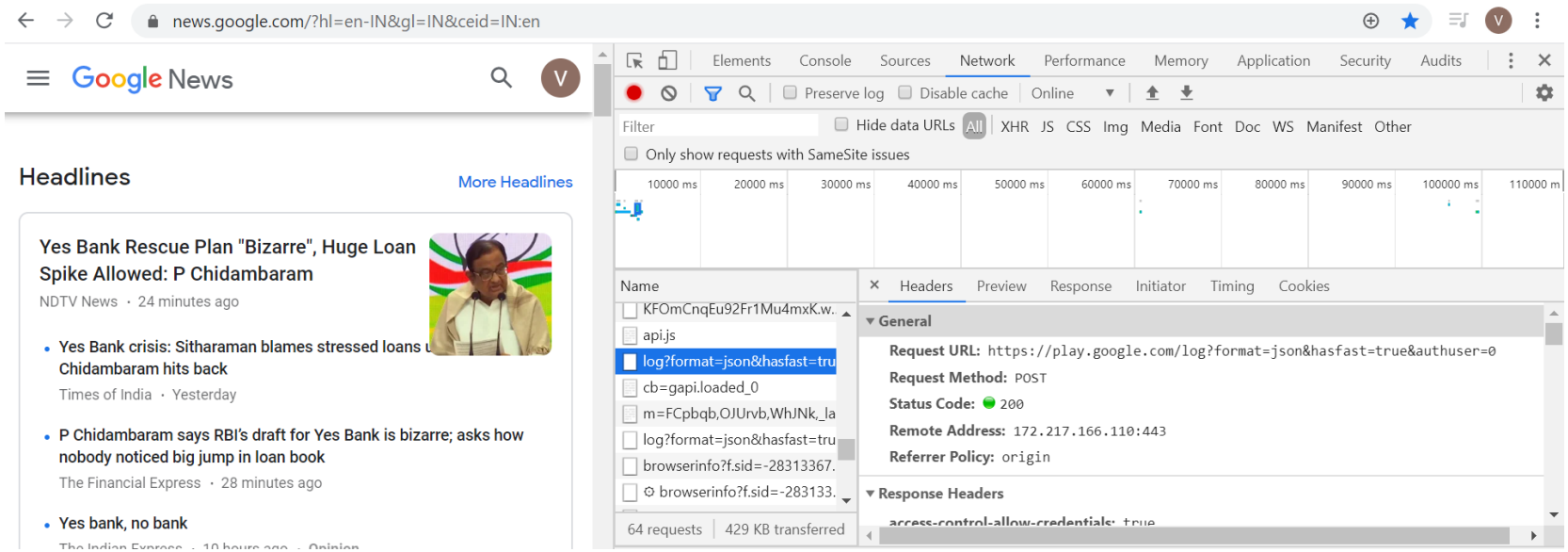
- Simple Scaling
  - Increase/Decrease the server instances
    - at specific times, number of concurrent users, etc.
    - by a specific count (say, 10% or 10 instances)
  - Apply a cooldown time period to let the new servers come into action
- Target Tracking
  - Attempts to keep a specific metric
    - For e.g., Keep Average CPU utilization at 50%
    - Increase server instances above threshold
    - Decrease the server instances below threshold
- Step Scaling
  - Improvement over simple scaling
  - Increase in steps – Say 2 instance at a time

AWS Dynamic Scaling Policies: https://docs.aws.amazon.com/autoscaling/ec2/userguide/scale-your-group.html

Scaling in AWS



Putting it all Together!



Summary