Session and Cookies

Web Dev, Spring 2021

HTTP is stateless

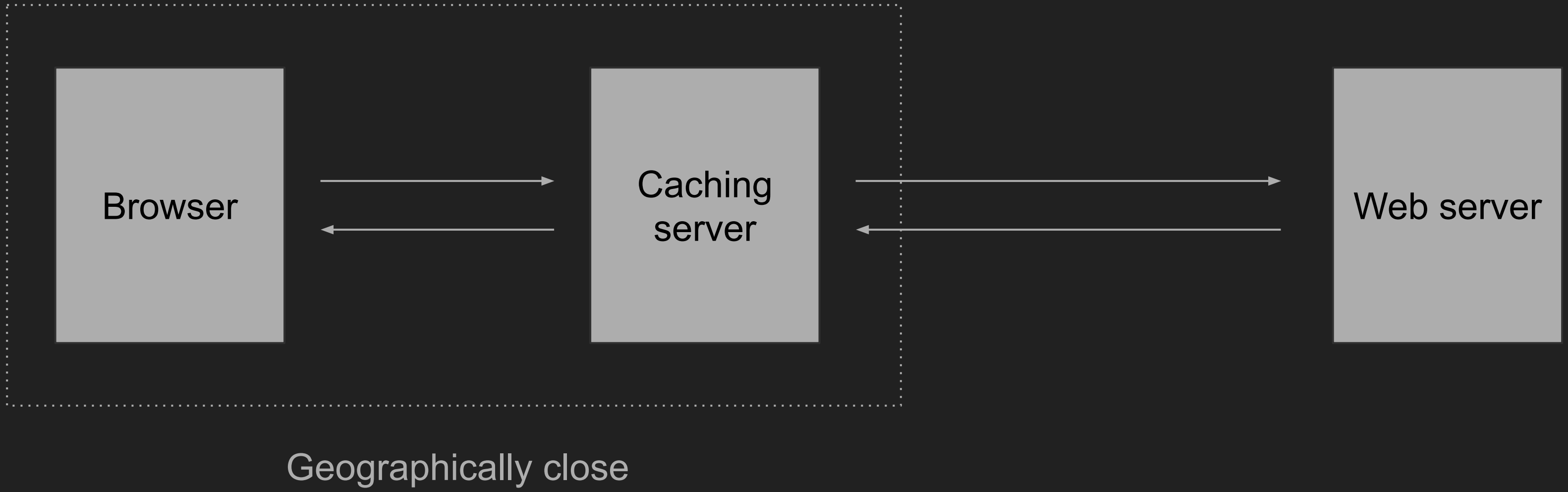
Every HTTP request is its own thing

- server should be able to respond to a request using only information contained in the request
- no "context" maintained between calls

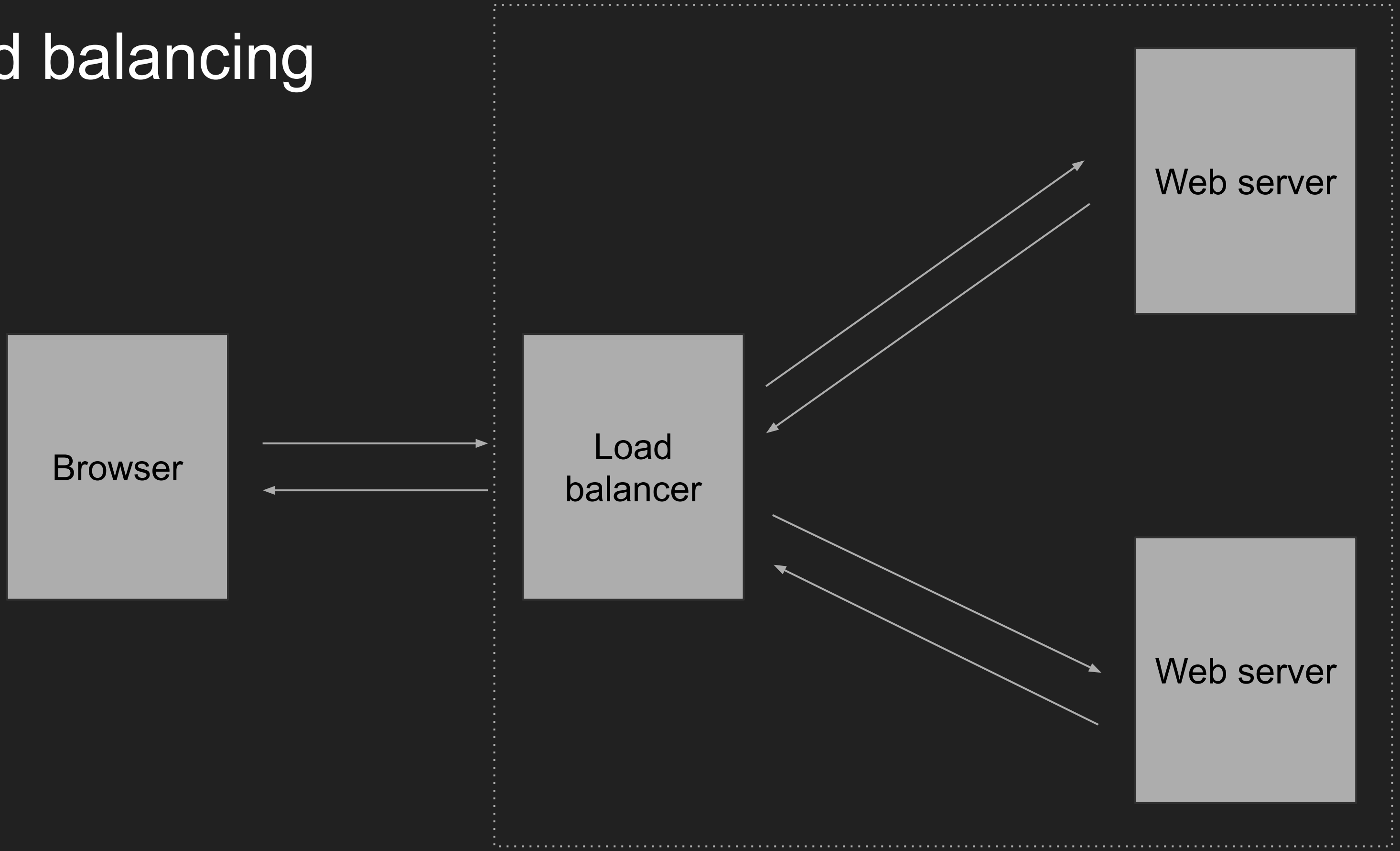
Advantages

- Can (mostly) cache responses to GET requests
- Can use load balancing servers

Caching



Load balancing



But… sessions!

Users are often interested in the notion of a session

- session = a span that represents a related sequence of interactions
- what defines a session is that state is maintained throughout the session
- e.g., shopping cart, logged-in status

HTTP does not maintain context (= state) between calls

So we need to maintain state "manually" at the application level

Cookies

A cookie is just some state associated with a domain (web server)

- a cookie has a name and a value (a string)

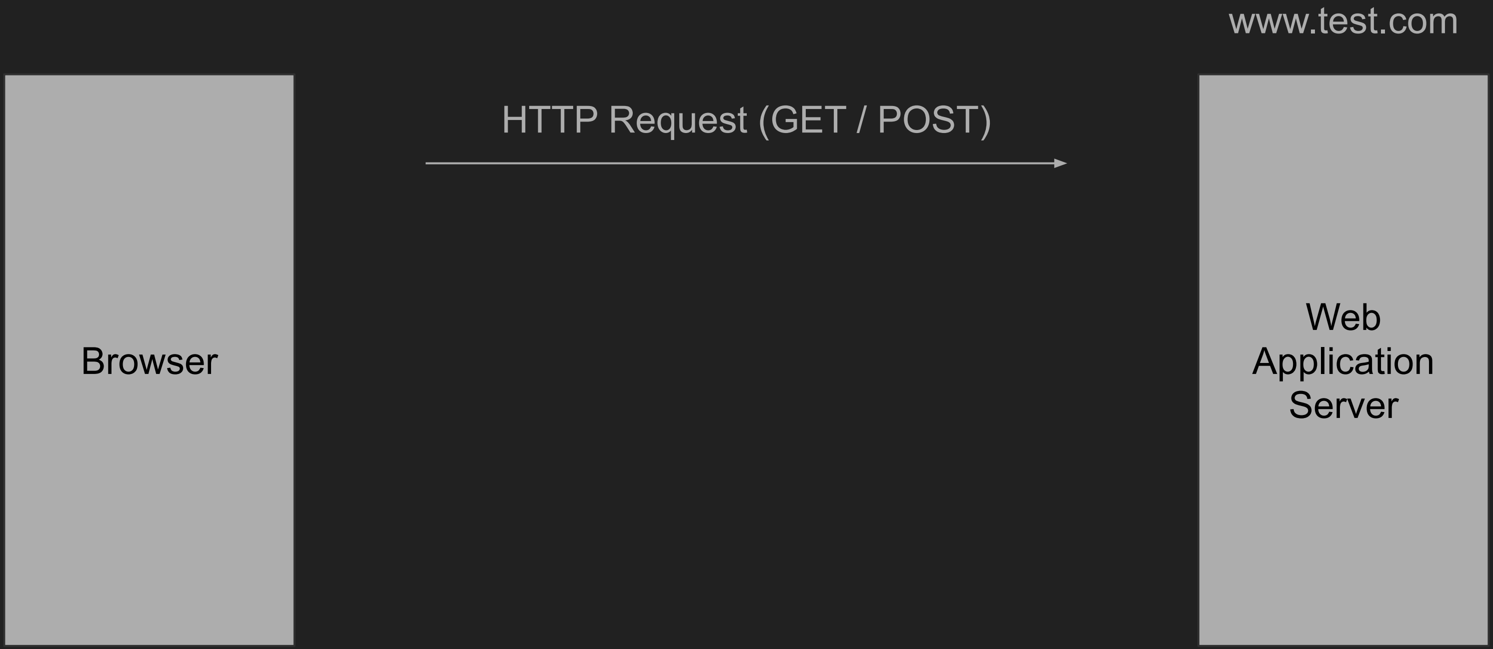
Created by a web server and sent to the client browser and kept in the browser

Sent to the web server with every request from the browser

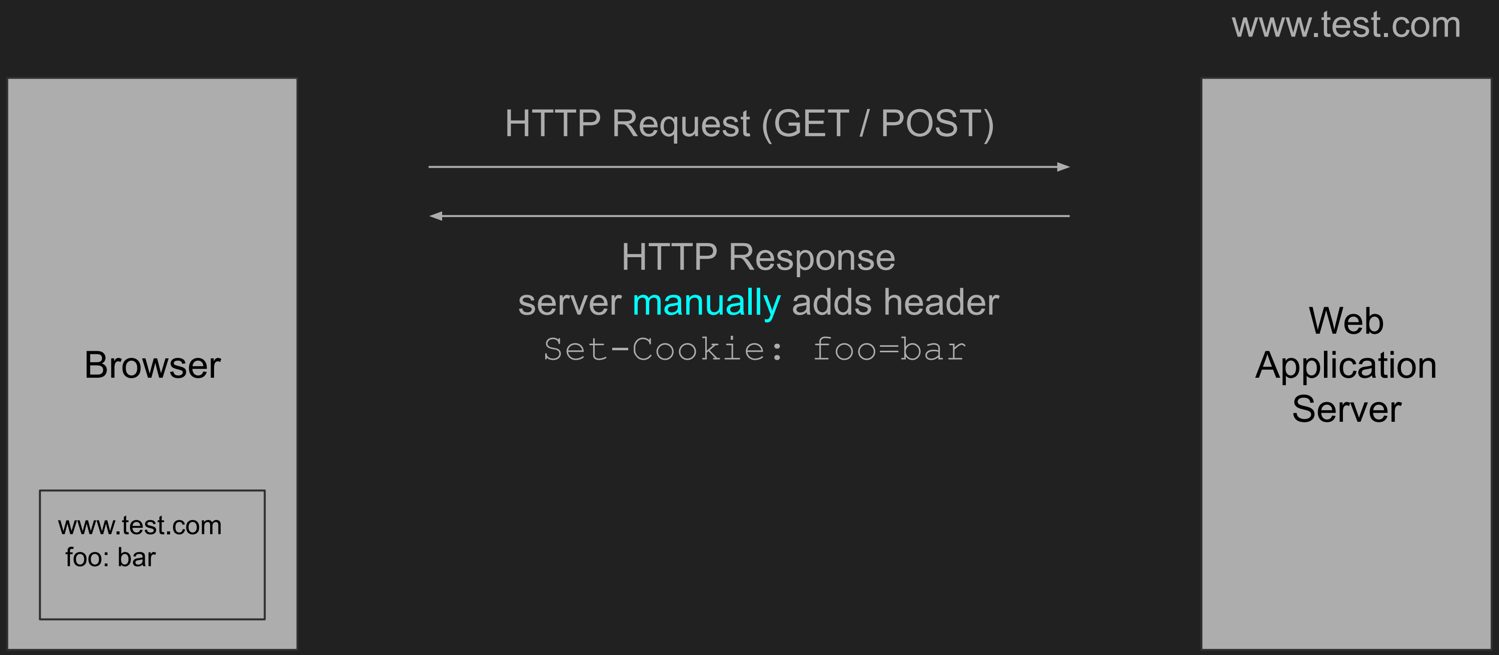
Web server may send new values for a cookie (= state update)

Advantage: HTTP requests carry their own context (= state)

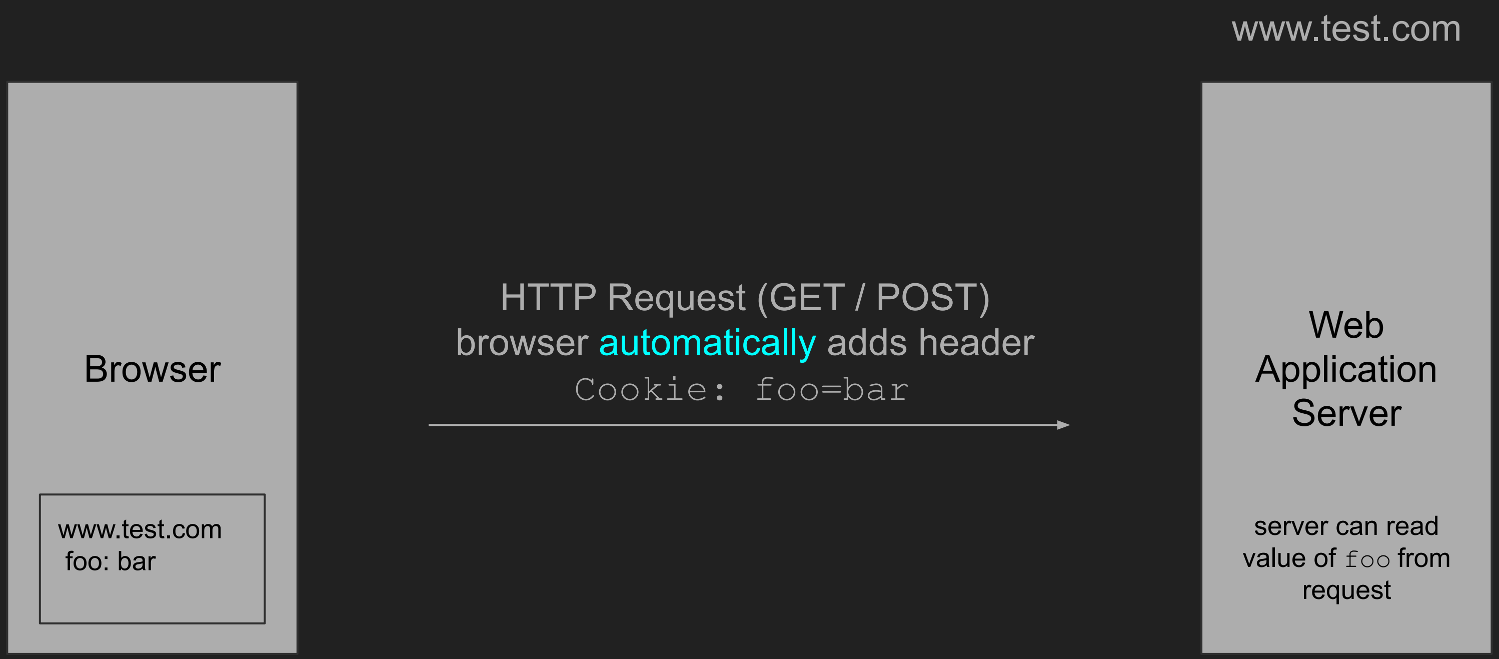
Cookies process



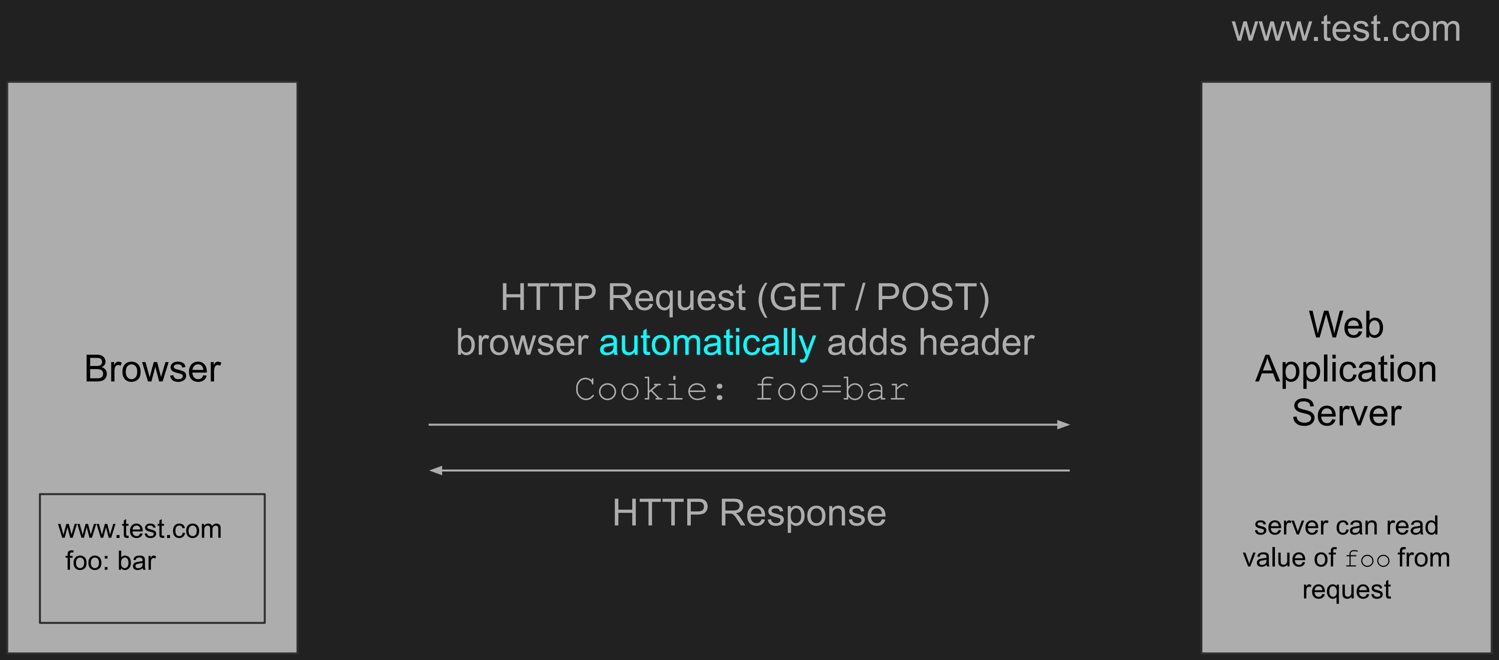
Cookies process



Cookies process


www.test.com
Browser
www.test.com
foo: bar
HTTP Request (GET / POST)
browser automatically adds header
Cookie: foo=bar
Web Application Server
server can read value of foo from request

Cookies process


www.test.com
Browser
www.test.com
foo: bar
HTTP Request (GET / POST)
browser automatically adds header
Cookie: foo=bar
HTTP Response
Web Application Server
server can read value of foo from request

Demo

Cookie for tracking history

Cookies limitations

Cookie:

- limited to 4 Kb
- limited to 50 cookies per server
- limited to 3000 cookies total
- potential for staleness
- storing state on the client may be a security risk (even if encrypted)

Better to store session data on the server

But… HTTP is stateless?

Storing session data on the server

Server holds session state in a dictionary

key → state

Server puts the key (a session ID) in a cookie

- client sends the session ID every time it sends an HTTP request
- server can access session state by looking up the session ID in the dictionary

Problem: doesn't work in load balanced scenario

Solution: store the dictionary in a database (Redis) running on another server