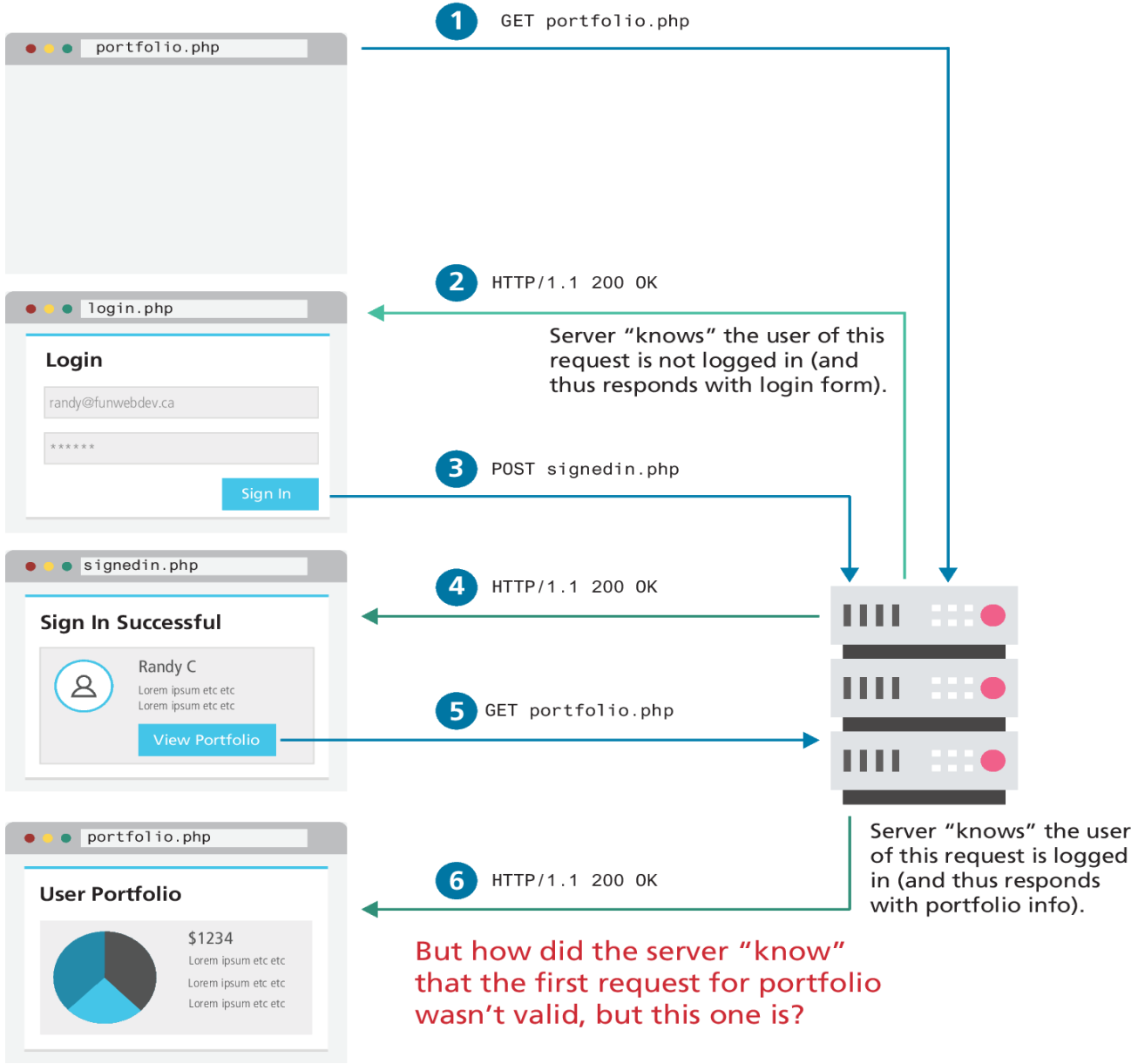
Session

Sridhar Alagar

Problem of state in WebApps

- Need to maintain information (state) across multiple requests



Session

- A **session** is a temporary stateful interaction between a client and a web server

Session state lookup problem in http

- HTTP requests are disconnected and does not carry information to uniquely identify a session

- Solution:
  - Include some info in the requests that tell us about the session

- In the context of HTTP, a session is a series of requests and responses that occur between a client and a server, where the state of the user is maintained across those requests

Where will the state be maintained?

- Client (Browser, local storage)
- Webserver (stateless?)
- Storage system

Early solution: Cookies

- State set by the server is attached to every request by the browser
- Response from server with state info (name/value pair) to be set in a cookie

```
HTTP/2.0 200 OK
Content-Type: text/html
Set-Cookie: yummy_cookie=chocolate; Max-Age=2592000
Set-Cookie: tasty_cookie=strawberry

[page content]
```

- Subsequent request message with state info from the cookie

```
GET /sample_page.html HTTP/2.0
Host: www.example.org
Cookie: yummy_cookie=chocolate; tasty_cookie=strawberry
```

Drawbacks of storing session state in cookies

- User sensitive information (login/password, etc.)
  - Can be protected using cryptography
  - But cookies can be deleted
- Size limitation
  - What if state is more than 4KB?
- Sending large of amounts of data over http increases bandwidth usage
  - Especially on the server as it may have handle large number of clients
- Ideally, server should have control over the state

Other options for storing session state

- Web server's memory
  - Fastest access to state
  - Size requirement can be an issue (large number of active users)
  - Makes load balancing across web servers hard
- Storage system
  - Easily shared across all webservers
  - Too much load on the storage system
  - Resilient storage system for temporary state could be an overkill
- Specialized storage system
  - Fast access for small, short-lived data
  - Many options: memcache, redis – in memory key-value store

Mapping requests to its session

- For a request in a session, how to get its session state?
  - Store a reference (session-id) to its session state
- Server sends the session-id with the response to the first request using set-cookie
- Subsequent request messages carry session-id in a cookie

Express-session

- A middleware for express to handle sessions

```
npm install express-session
```

- Stores session-id safely in a cookie
- Stores state in a session store, by default in server's memory
- Creates and fetches session state for request handlers

Express-session – login example

- Login handler stores user info in **`req.session.user`**
- All other handlers read **`req.session.user`**
  - If not set, redirect to login page
  - Otherwise, know the logged in user
- Can put other session info in **`req.session`**
- Logout handler should destroy the session

  `req.session.destroy()`

MongoDB Session store – connect-mongo

Basic usage

```
const session = require('express-session');
const MongoStore = require('connect-mongo’);


app.use(session({
  store: MongoStore.create({ mongoUrl: 'mongodb://localhost/test-app' })
}));
```

Web Storage – alternate to cookie storage

- Provides mechanism for browsers to store key/value pair
- sessionStorage – per origin store available for duration of page session
- localStorage – per origin store persists even after the browser is closed
- Larger storage space up to 10 MB
- At the client side – so other issues of cookies remain

Sources

1. express-session
2. MDN - Using HTTP Cookies
3. CS142 Lectures
4. Fundamentals of Web Development. 3rd Edition by Randy Connolly and Ricardo Hoar.
5. ChatGPT