



THE UNIVERSITY  
of ADELAIDE



CRICOS PROVIDER 00123M

Lecture 31 – Cookies and Sessions

# Web & Database Computing

[adelaide.edu.au](http://adelaide.edu.au)

*seek* LIGHT

# Cookies

- Recall that HTTP is *stateless*
  - Each request is unrelated to any previous requests
  - If the server dies, it doesn't need to recover any information about previous requests when it comes back up
- But sometimes we **need** state – how can we address this problem?
  1. On the server, we can store state in databases, but how do we relate this information to new requests? For example if a user has a shopping cart and we want to show them the contents that was in their cart when they last visited our page.
  2. What if we want to store short term information (only since the user logged in) that we don't want to store long term in the database. For example, pages visited since logging in?

# Cookies

- To solve this problem, we could make users login to identify themselves, but then we can't have anonymous shopping or browsing.
- **Cookies** were created to allow the server (e.g. express) to store state on the client (e.g. browser) that will then be sent with all future requests to that server.
- So instead of making the user log in, we'll give them a cookie.
- Lots of websites use cookies. Have a look in your browser settings or in your web inspector tool and you can see the cookies that have been set by different web sites. Note the information and characteristics (domain, expiration,...)

# Adding a cookie in express

- As we have seen many times before, there are usually modules for anything we want to do in express.
- Express Module for Cookies
  - `cookie-parser` (already included in the code that is created for you by express generator, have a look in app.js)
- We can create cookies on the client by setting a cookie on the response at the server and then sending the response to the client. This is built in to express.  
`res.cookie('cookie_name', 'cookieinfo');`  
`res.send(200);` or `res.json(jsonValue);` or `res.render('mypage');`
- Future requests from the client will include the cookie. The server can get the cookie values with:  
`req.cookies.cookie_name`

# Demo

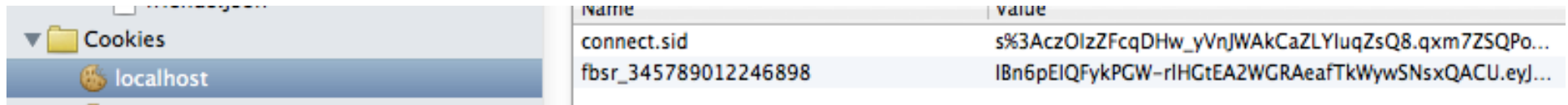
- Some references for Cookies in express:
  - basic tutorial  
<https://www.codementor.io/nodejs/tutorial/cookie-management-in-express-js>
  - API for reading cookies  
<https://github.com/expressjs/cookie-parser>
  - API for setting cookies  
<http://expressjs.com/en/4x/api.html#res.cookie>

# Sessions

- Back to problem 2. Assume a user has logged in. How do we track information about the user since logging in?
- We could just store all the data we want to keep as cookies which expire when the browser is closed.
  - All of this information is visible to the user – they can look at their cookies.
  - All of this information has to be sent to the server with every request.
- This can expose information about the internal workings of your web application (product id numbers, user id numbers, etc) which can be a security risk – if I edit the cookie and change the user id can I get someone else's information?
- **Sessions** allow this temporary user information to be stored at the server instead of the client.

# How does this work?

- The sessions software sets a cookie at the client browser with a session id.



The image shows a browser's cookie manager interface on the left, displaying a folder named 'Cookies' and a sub-entry for 'localhost'. To the right is a table with two columns: 'Name' and 'Value'. The table contains two rows of session data.

Name	Value
connect.sid	s%3AczOlzZFcqDHw_yVnJWAKCaZLYluqZsQ8.qxm7ZSQPo...
fbsr_345789012246898	IBn6pElQFykPGW-rlHGtEAZWGRAeafTkWywSNsxQACU.eyJ...

- Whenever the browser calls the server, it sends this cookie
- The session uses the session id (connect.sid in this example) to get the right session data for that id.
- Sessions will end and the information will be removed whenever the browser or the server shutdown, whichever happens first.



# Sessions in express

- Setting up
  - Need the express-session module, require in app.js  
`var session = require('express-session')`
  - Create a session middleware to handle sessions in app.js  
`app.use(session({  
                    secret: 'a string of your choice',  
                  }))`
- Using sessions
  - The session middleware will automatically create the sessions for you and store information in a variable `req.session` You can set and read session variables using `req.session` in your routes in index.js  
`req.session.variableName = 5;  
console.log(req.session.variableName);`
- It is also possible to store your sessions in your database to make them survive the server shutting down
  - <https://www.npmjs.com/package/express-mysql-session>
  - This is beyond the scope of our course, but provided for the curious.



# Demo

- References:
- API <https://github.com/expressjs/session>
- Tutorial  
<https://codeforgeek.com/2014/10/express-complete-tutorial-part-4/>

# Cookies or Sessions?

	Cookies	Sessions
expiration	Can be set (session, years)	When browser is closed
Where data is stored	On the client	On the server (session id on the client)
size	Limited by browser – generally 4093 bytes total (all cookies from a given site)	Limited only by memory/storage



THE UNIVERSITY  
*of* ADELAIDE

