Server Side

express().listen(80)
<table>
<thead>
<tr>
<th>C</th>
<th>post</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>get</td>
</tr>
<tr>
<td>U</td>
<td>put</td>
</tr>
<tr>
<td>D</td>
<td>delete</td>
</tr>
</tbody>
</table>
Best Practices

• think about “resources”
• elements & collections
• map out the 4 methods for each
• Prefer Nouns, Plural, Concrete
• Use Parameters for more advanced queries
Express

- routes
- static files
- middleware
- template engine - ejs
- method override
Session Management

HTTP is stateless
Express

- Routes are served by a callback

```javascript
app.post("/courses", (req, res) => {
  db.add({
    id: req.body.code,
    when: new Date(),
    what: req.body.what,
    who: req.body.who
  });
  res.redirect("/courses")
});
```

- Resources can be static or dynamically generated

```javascript
app.get("/courses/csc404", (req, res) => {
  res.send(`<html>
    <body>
      <p>This is not a REAL course</p>
    </body>
  </html>
`);
});
```
Express

• Express provides a middleware architecture

```javascript
function myLogger(req, res, next) {
  console.log('LOG:', req.method, req.url, req.body)
  res.remember = req.url
  next()
}

app.use(myLogger)
```
Cookies

- HTTP is stateless
- HTTP cookies provide illusion of state!
  - Managed by server
  - Stored by browser