Sessions in Hapi

Agenda

- Simple precursor to Sessions
- Sessions via Cookies in Hapi

Sharing Information across an App

- Before server launches, ‘bind’ an array of donations to the server object.
- Most commonly used to share database connection information

index.js

```
...
server.bind({
  donations: [],
});
...
```

Sharing Information across an App

- This ‘donations’ array can subsequently be accessed in all handlers.
- Each handler can read/write to this shared data structure
- All users donations held in an array in memory

index.js

```
...
server.bind({
  donations: [],
});
...
```

```
exports.donate = {

  handler: function (request, reply) {
    const data = request.payload;
    this.donations.push(data);
    reply.redirect('/report');
  },

};
```

donations.js

Separating out User Donations

- Try to keep track of
  - all users,
  - the current user
  - all donations.

index.js

```
...
server.bind({
  currentUser: {},
  users: {},
  donations: [],
});
...
```

Registering & Authenticating Users

```
signup: {
  handler: function(request, h) {
    const user = request.payload;
    this.users[user.email] = user;
    this.currentUser = user;
    return h.redirect('/home');
  }
},


login: {
  handler: function(request, h) {
    const user = request.payload;
    if ((user.email in this.users) && (user.password === this.users[user.email].password)) {
      this.currentUser = this.users[user.email];
      return h.redirect('/home');
    }
    return h.redirect('/');
  }
},
;
```

- Record user object at registration
- Record current user at login

accounts.js

Creating & Listing Donations

```
donate: {
  handler: function(request, h) {
    const data = request.payload;
    data.donor = this.currentUser;
    this.donations.push(data);
    return h.redirect('/report');
  }
}
```

donations.js

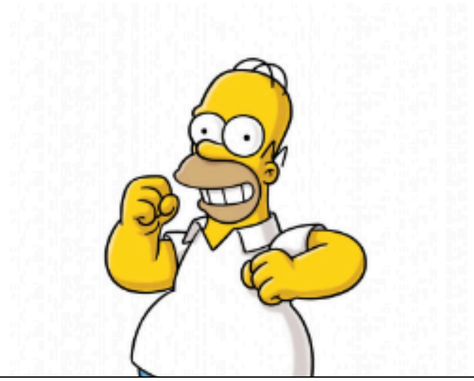
- Record donation + donor when creating donation

```
report: {
  handler: function(request, h) {
    return h.view('report', {
      title: 'Donations to Date',
      donations: this.donations
    });
  }
},
```

donations.js

- Send all donations to the view

Donation | Donate | Report | Logout



Amount	Method donated	Donor
100	paypal	homer simpson
50	direct	homer simpson
50	paypal	homer simpson

```
...
<tbody>
  {{#each donations}}
    <tr>
      <td> {{amount}} </td>
      <td> {{method}} </td>
      <td> {{donor.firstName}} {{donor.lastName}} </td>
    </tr>
  {{/each}}
</tbody>
...
```

donationlist.hbs

Summary

- Current approach - brittle and not scalable
  - Server.bind to maintain global data
  - Store user + donation data structures
- Revised Approach
  - Migrate to more robust, cookie based session management
  - Introduce proper persistence capability (a database)

Sessions

- HTTP is described as a stateless protocol - every new request is just as anonymous as the last.
- This sounds very unhelpful for a protocol that powers websites, where users expect to be remembered as they go to page to page.
- Cookies to the Rescue:
  - A request comes to a web application with a cookie
  - using the cookie the server can look up information about the user, either from the cookie itself or from server-side storage.
  - It can then forget all about them for a while, until the next request and the same process continues over for every request.

hapi-auth-cookie

- A Hapi Plugin to manage cookie access and management.
- Must be downloaded, installed and registered (like all plugins)

https://github.com/hapijs/hapi-auth-cookie

hapi-auth-cookie

hapi Cookie authentication plugin

build failing

Lead Maintainer: Eran Hammer

Cookie authentication provides simple cookie-based session management. The user has to be authenticated via other means, typically a web form, and upon successful authentication the browser receives a reply with a session cookie. The cookie uses Iron to encrypt and sign the session content.

Subsequent requests containing the session cookie are authenticated and validated via the provided `validateFunc` in case the cookie's encrypted content requires validation on each request.

It is important to remember a couple of things:

1. Each cookie operates as a bearer token and anyone in possession of the cookie content can use it to impersonate its true owner.
2. Cookies have a practical maximum length. All of the data you store in a cookie is sent to the browser. If your cookie is too long, browsers may not set it. Read more here and here. If you need to store more data, store a small amount of identifying data in the cookie and use that as a key to a server-side cache system.

The `'cookie'` scheme takes the following options:

- `cookie` - the cookie name. Defaults to `'sid'`.
- `password` - used for Iron cookie encoding. Should be at least 32 characters long.
- `ttl` - sets the cookie expires time in milliseconds. Defaults to single browser session (ends when browser closes). Required when `keepAlive` is `true`.
- `domain` - sets the cookie Domain value. Defaults to none.
- `path` - sets the cookie path value. Defaults to `/`.
- `clearInvalid` - if `true`, any authentication cookie that fails validation will be marked as expired in the response and cleared. Defaults to `false`.
- `keepAlive` - if `true`, automatically sets the session cookie after validation to extend the current session for a new `ttl` duration. Defaults to `false`.

hapi-auth-cookie Installation & Registration

npm install command

```
npm install hapi-auth-cookie
```

package.json updated

```
{
  "name": "donation-web",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "handlebars": "^4.0.12",
    "hapi": "^18.0.0",
    "hapi-auth-cookie": "^9.1.0",
    "inert": "^5.1.2",
    "vision": "^5.4.4"
  },
  "devDependencies": {
    "prettier": "^1.16.0"
  },
  "prettier": {
    "singleQuote": true,
    "printWidth": 120
  }
}
```

Register in index.js

```
  await server.register(require('hapi-auth-cookie'));
```

hapi-auth-cookie Configuration

- Set an auth ‘strategy’ before application starts
- Specifies range or parameters, including:
  - password for securing cookie
  - cookie name
  - time to live (expiry)

```
...
server.auth.strategy('standard', 'cookie', {
  password: 'secretpasswordnotrevealedtoanyone',
  cookie: 'donation-cookie',
  ttl: 24 * 60 * 60 * 1000,
});
...
```

hapi-auth-cookie Configuration

- By default hapi-auth-cookie will only allow the cookie to be transferred over a secure TLS/ SSLconnection.
- This may not be convenient during development so you can set the isSecure option to false.
- Set ‘standard’ as the default strategy for all routes

```
...
server.auth.strategy('standard', 'cookie', {
  password: 'secretpasswordnotrevealedtoanyone',
  cookie: 'donation-cookie',
  isSecure: false,
  ttl: 24 * 60 * 60 * 1000,
});

server.auth.default({
  strategy: 'standard',
});

...
```

Annotating Routes

- All routes are now ‘guarded’ by default, cookie based authentication mechanism
- Any attempt to visit a route will be rejected unless valid cookie detected.
- Some routes need to be available (to signup or login for instance)
- These routes must specifically disable auth mechanism

```
...

server.auth.default({
  mode: 'required',
  strategy: 'standard'
});

...
```

```
...
index: {
  auth: false,
  handler: function(request, h) {
    return h.view('main', { title: 'Welcome to Donations' });
  }
},
showSignup: {
  auth: false,
  handler: function(request, h) {
    return h.view('signup', { title: 'Sign up for Donations' });
  }
},
...
```

Setting the Cookie

- Set the cookie if correct user credentials presented.

```
request.cookieAuth.set({ id: user.email })
```

```
...
login: {
  auth: false,
  handler: function(request, h) {
    const user = request.payload;
    if (user.email in this.users && user.password === this.users[user.email].password) {
      request.cookieAuth.set({ id: user.email });
      return h.redirect('/home');
    }
    return h.redirect('/');
  }
},

...
```

Reading the Cookie

- If cookie set, it can be read back in any handler
- We are storing logged in users email in this example
- Use this email to look up user details in some storage infrastructure (database).

```
request.cookieAuth.set({ id: user.email })
```

```
var donorEmail = request.auth.credentials.id;
```

```
donate: {
  handler: function(request, h) {
    const data = request.payload;
    var donorEmail = request.auth.credentials.id;
    data.donor = this.users[donorEmail];
    this.donations.push(data);
    return h.redirect('/report');
  }
}
```

Clearing the Cookie

- Cookie deleted
- Any attempt to access protected routes rejected

```
logout: {
  handler: function(request, h) {
    request.cookieAuth.clear();
    return h.redirect('/');
  }
}
```

Redirects

If route protected, and cookie deleted/ timed out

Redirect to login

```
server.auth.strategy('standard', 'cookie', {
  password: 'secretpasswordnotrevealedtoanyone',
  cookie: 'donation-cookie',
  isSecure: false,
  ttl: 24 * 60 * 60 * 1000,
  redirectTo: '/'
});
```

Cookies can be Inspected in Browser

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network | Storage | Accessibility

- Cache Storage
- Cookies
  - http://localhost:3000
- Indexed DB
- Local Storage
- Session Storage

Filter items

Name	Domain	Path	Expires on	Last accessed on	Val
donation-web	localhost	/	Thu, 31 Jan 2019 07:01:1...	Wed, 30 Jan 2019 07:01:...	Fe26.2**

Filter values

Data

```
donation-web: "Fe26.2**78d4c93c186076e4cd...LNHdXmQBk2v4cTgKTZKEKvBWU"
  CreationTime: "Wed, 30 Jan 2019 07:01:14 GMT"
  Domain: "localhost"
  Expires: "Thu, 31 Jan 2019 07:01:14 GMT"
  HostOnly: true
  HttpOnly: true
  LastAccessed: "Wed, 30 Jan 2019 07:01:14 GMT"
  Path: "/"
  Secure: false
  sameSite: "Strict"
```

Parsed Value

```
donation-web: Array
  0: "Fe26.2"
  1: ""
  2: "78d4c93c186076e4cd271c2b1d33e5a1548cf79134789aaee69cd0a4389b7174"
  3: "EV_4-Ds7qYD3Qqz3YxyPIQ"
  4: "TtPdRqA3xzll5DlvBJjXG3nr9gMPiXx_2ThreofArK-7HMKL18neAgUCrmibC99K"
  5: ""
  6: "28a6a80b7dece305e6cfa1c7661d50cbfe1220f121e1c5687c7a72289c405ebc"
  7: "zTN-pcJx0oYtzS4oQhLNHdXmQBk2v4cTgKTZKEKvBWU"
  length: 8
  __proto__: Array
```