Beaconator Readme

Hapi Web and API Server, with frontend dashboard. Based on Hapi Dash, which is based on Hapi Ninja.

TODO:

- /servers/api/server.js & /servers/gui/config/plugins.js : set up process monitoring with `good` module?

Local Setup

Prerequisites

- **Node.js**
- **Mongo** - official install instructions
- **pm2** - Run `npm install -g pm2`
- **gulp** - Run `npm install -g gulp`

Clone the repo:

```
git@bitbucket.org:fusionary/beaconator-web-app.git
cd beaconator-web-app
```

In the project's root directory:

```
npm install
npm run config
```

- **Answer the prompts** that follow. This will create two new unsourced files in the project's root: config.js and config.pm2.json
- **Update config.js** with proper user name/passwords on the wiki

Set up mongo:

- Type `mongo` on the command line to enter the mongo cli. Then:
- Create "userAdmin" user:

```
use admin
db.createUser({user: 'admin',pwd: '[somePassword]', roles: [ { role: 'userAdminAnyDatabase', d
```

- Create user for beacon db:

```
mongo --port 27017 -u admin -p [somePassword] --authenticationDatabase admin

db.createUser({user: "fusionary",pwd: "password-that-is-in-config.js",roles: [ { role: "readWr
```

Local Development

- Run `npm start` to start the Hapi server and Mongodb (`npm stop` to stop)
- Run `gulp serve` to start live realoading of changed files (c to stop)
- Run `npm run restart` to manually restart any currently running Hapi servers ** Notice the `run` after `npm` there.** That's because "restart" isn't one of the default npm scripts.

Authentication

There is separate authentication for the different servers in the app. Once a user registers they will be created with a random API token. This is used to access the API.

API

The API uses Hawk authentication. There are two auth strategies pre-configured, 'core' and 'web'. Core is for core functionality, i.e. internal API endpoints. Core credentials are hardcoded and should be changed before deployment. Web is for web-facing endpoints, i.e. for registered users. Core credentials will work for web routes.

```
// EXAMPLE
// Only allow core to use route (ie internal)
var routeConfig = {
    auth: 'core',
    ...
}

// Only allow registered users to use route
var routeConfig = {
    auth: 'web',
    ...
}
```

GUI

The GUI uses Cookie authentication.

```
// EXAMPLE: add session auth to hapi route
// Only allow registered users to use route
var routeConfig = {
    auth: 'session',
    ...
}
```

Credits

- Simon Maxwell-Stewart, Saul Maddox, and all of the open source code that people have made available.
- The dashboard uses the free Dashgum responsive theme, design by Carlos Alvarez. Framework used: Bootstrap 3.2