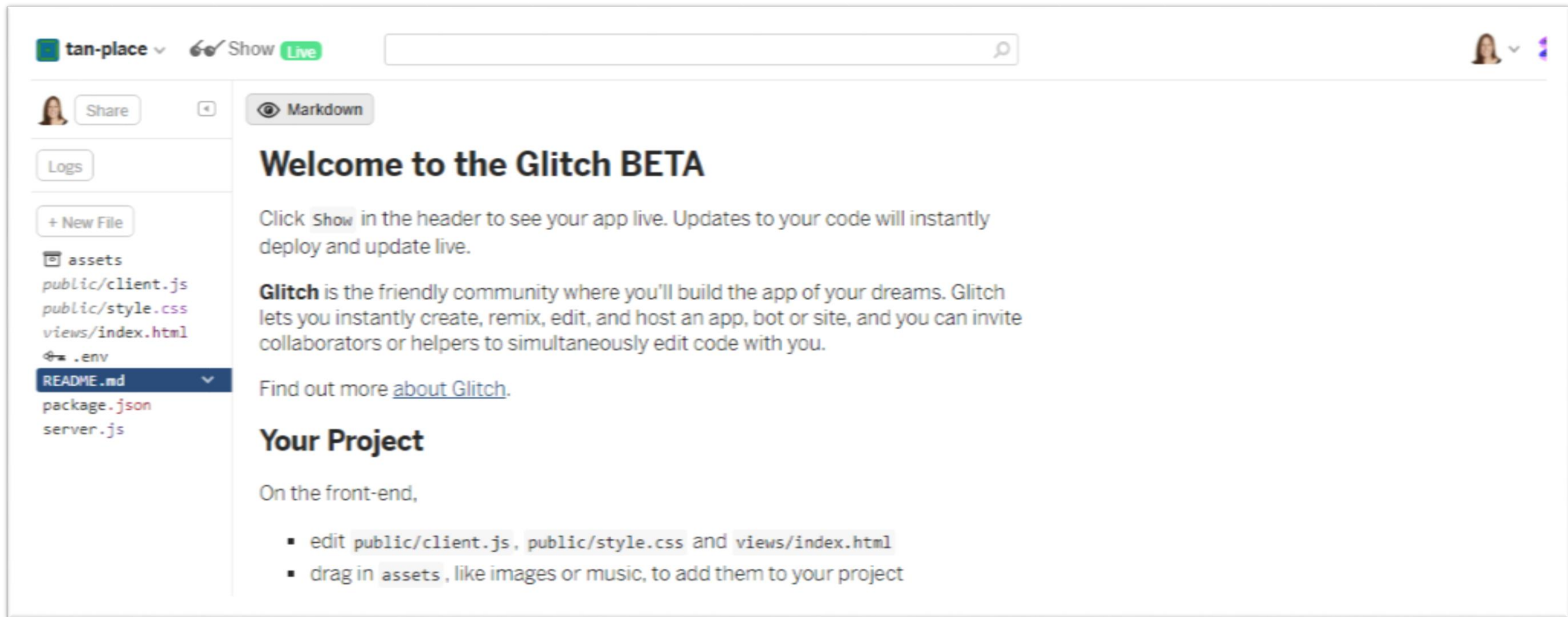


Glitch Tour

Prerequisite tools on your Workstation

- none!
- (apart from a browser + a github account)



- First screen is the “source” for a running, live web project

The screenshot shows a Glitch project interface. At the top, the project name 'tan-place' is displayed next to a user profile icon and a 'Show Live' button. Below the project name is a 'Share' button and a 'Logs' button. A '+ New File' button is also present. The file explorer on the left lists files: 'assets', 'public/client.js', 'public/style.css', 'views/index.html', '.env', 'README.md' (highlighted), 'package.json', and 'server.js'. The main content area has a 'Markdown' toggle and a 'Welcome to the Glitch BETA' heading. It includes instructions on how to use the 'show' button and a description of Glitch as a community for building apps. Below this is a 'Your Project' section with instructions for front-end and back-end development. A 'New!' badge is visible in the bottom right corner.

tan-place Show **Live**

Share

Logs

+ New File

assets

public/client.js

public/style.css

views/index.html

.env

README.md

package.json

server.js

Welcome to the Glitch BETA

Click **show** in the header to see your app live. Updates to your code will instantly deploy and update live.

Glitch is the friendly community where you'll build the app of your dreams. Glitch lets you instantly create, remix, edit, and host an app, bot or site, and you can invite collaborators or helpers to simultaneously edit code with you.

Find out more [about Glitch](#).

Your Project

On the front-end,

- edit `public/client.js`, `public/style.css` and `views/index.html`
- drag in `assets`, like images or music, to add them to your project

On the back-end,

- your app starts at `server.js`

New!

Project name
(automatically
generated)


Link to
running app
(to share)

Files in
the
project


Current
File
(editable)

Link to your
Profile

Link to
Community,
resources,
options

 tan-place

Show Live

 Share

Logs

+ New File

assets

public/client.js

public/style.css

views/index.html

.env

README.md

package.json

server.js

Markdown

Welcome to the

Click `show` in the header will instantly deploy and

Glitch is the friendly community for building your dreams. Glitch lets you create any web app, bot or site, and you can share it with others. You can simultaneously edit code with others.

Find out more [about Glitch](#)

Your Project

On the front-end,

- edit `public/client.js`
- edit `views/index.html`
- drag in `assets`, like images

On the back-end,

- your app starts at `server.js`

Secure | https://tan-place.glitch.me

Apps Waterford Institute of Google Virtual Learning

A Dream of the Future

Oh hi,

Tell me your hopes and dreams:

- Find and count some sheep
- Climb a really tall tree
- Wash the dishes

[Remix this in Glitch](#)

- Project is always running live (provided there are no source errors)


Project Structure

- Glitch projects not just web sites!
- They are web apps, divided into:
 - Front-end files
 - Back-end files

A screenshot of a file explorer interface. It shows a list of files and folders. At the top is a folder icon followed by 'assets'. Below it are three files: 'public/client.js', 'public/style.css', and 'views/index.html'. Then there is a key icon followed by '.env'. Below that is a dark blue bar with 'README.md' in white text and a white downward arrow on the right. Below the bar are two more files: 'package.json' and 'server.js'.

```
assets
public/client.js
public/style.css
views/index.html
.env
README.md
package.json
server.js
```


Front End



```
assets  
public/client.js  
public/style.css  
views/index.html
```

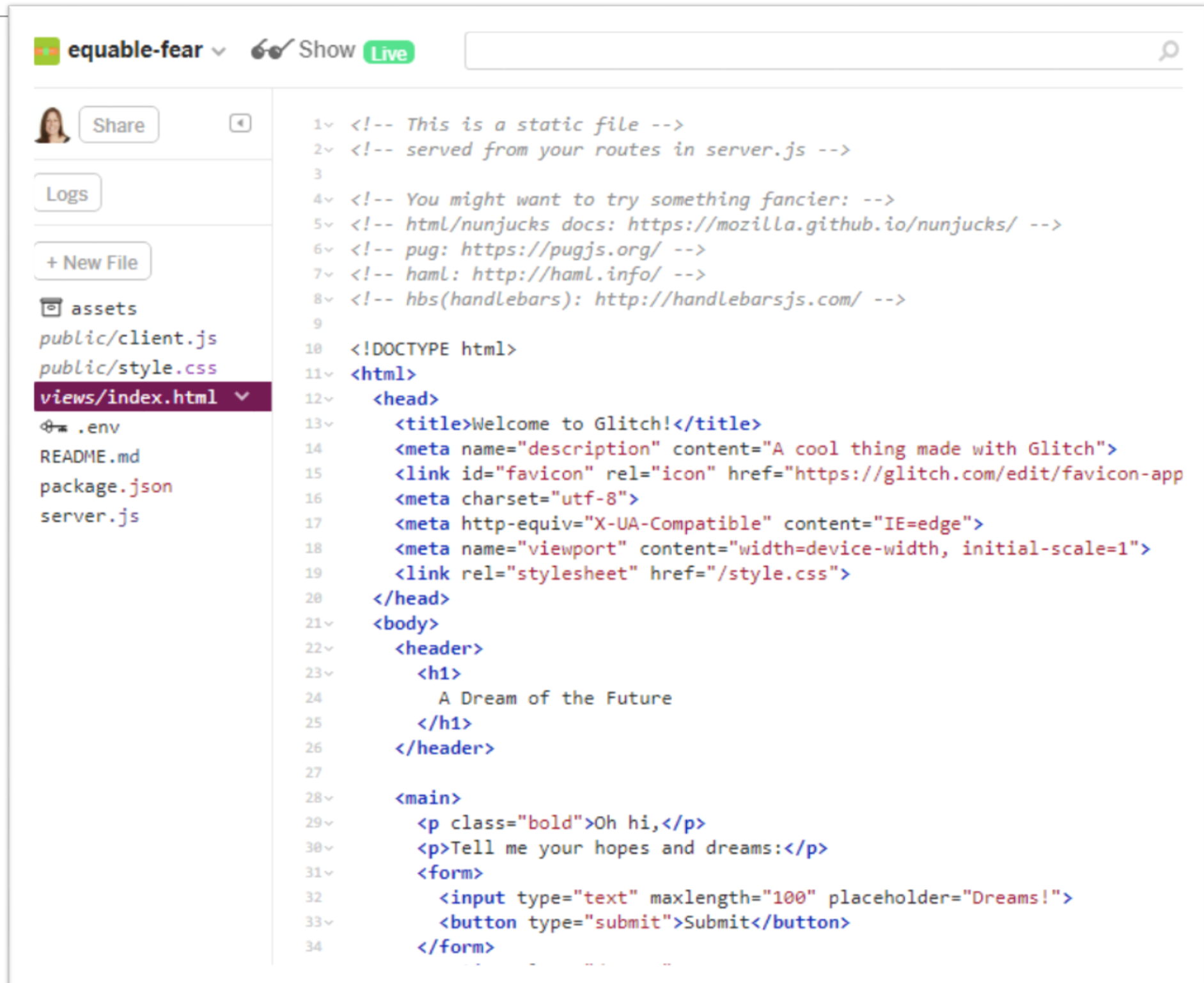
- Comparable to the web site you developed in previous module.
- html files + stylesheets + images
- Templating also possible.
- Also, access to the server side is implicit.
- This means you can build apps that have behaviour + state (much more on this later)

Back end



- An application - written in javascript - and hosted in the cloud.
- Many types of application supported.
- We will focus on Javascript applications written using node.js
- This is the default toolkit for Glitch.

The Starter App



The screenshot shows a Glitch project editor interface. At the top, the project name is "equable-fear" with a dropdown arrow, followed by a "Show" button and a "Live" button. A search bar is on the right. On the left sidebar, there's a "Share" button, a "Logs" button, and a "+ New File" button. Below these, a file tree shows "assets", "public/client.js", "public/style.css", "views/index.html" (selected), ".env", "README.md", "package.json", and "server.js". The main editor area displays the content of "views/index.html" with line numbers 1 through 34. The code is a static HTML file with comments and a form.

```
1<!-- This is a static file -->
2<!-- served from your routes in server.js -->
3
4<!-- You might want to try something fancier: -->
5<!-- html/nunjucks docs: https://mozilla.github.io/nunjucks/ -->
6<!-- pug: https://pugjs.org/ -->
7<!-- haml: http://haml.info/ -->
8<!-- hbs(handlebars): http://handlebarsjs.com/ -->
9
10<!DOCTYPE html>
11<html>
12  <head>
13    <title>Welcome to Glitch!</title>
14    <meta name="description" content="A cool thing made with Glitch">
15    <link id="favicon" rel="icon" href="https://glitch.com/edit/favicon-app
16    <meta charset="utf-8">
17    <meta http-equiv="X-UA-Compatible" content="IE=edge">
18    <meta name="viewport" content="width=device-width, initial-scale=1">
19    <link rel="stylesheet" href="/style.css">
20  </head>
21  <body>
22    <header>
23      <h1>
24        A Dream of the Future
25      </h1>
26    </header>
27
28    <main>
29      <p class="bold">Oh hi,</p>
30      <p>Tell me your hopes and dreams:</p>
31      <form>
32        <input type="text" maxlength="100" placeholder="Dreams!">
33        <button type="submit">Submit</button>
34      </form>
```

The Starter App

← → ↻ Secure | https://tan-place.glitch.me

Apps Waterford Institute of Google Virtual Learning

A Dream of the Future

Oh hi,

Tell me your hopes and dreams:

Submit

- Find and count some sheep
- Climb a really tall mountain
- Wash the dishes

[Remix this in Glitch](#)

equable-fear Show Live

Share

Logs

+ New File

assets

```
1 <!-- This is a static file -->
2 <!-- served from your routes in server.js -->
3
4 <!-- You might want to try something fancier: -->
5 <!-- html/nunjucks docs: https://mozilla.github.io/nunjucks/ -->
6 <!-- pug: https://pugjs.org/ -->
7 <!-- haml: http://haml.info/ -->
8 <!-- hbs(handlebars): http://handlebarsjs.com/ -->
9
10 <!DOCTYPE html>
11 <html>
12   <head>
13     <title>Welcome to Glitch!</title>
14     <meta name="description" content="A cool thing made with Glitch">
15     <link id="favicon" rel="icon" href="https://glitch.com/edit/favicon-app
16     <meta charset="utf-8">
17     <meta http-equiv="X-UA-Compatible" content="IE=edge">
18     <meta name="viewport" content="width=device-width, initial-scale=1">
19     <link rel="stylesheet" href="/style.css">
20   </head>
21   <body>
22     <header>
23       <h1>
24         A Dream of the Future
25       </h1>
26     </header>
27
28     <main>
29       <p class="bold">Oh hi,</p>
30       <p>Tell me your hopes and dreams:</p>
31       <form>
32         <input type="text" maxlength="100" placeholder="Dreams!">
33         <button type="submit">Submit</button>
34       </form>
```

A Dream of the Future

Oh hi,

Tell me your hopes and dreams:

- Find and count some sheep
- Climb a really tall mountain
- Wash the dishes

[Remix this in Glitch](#)

```
<body>
  <header>
    <h1>
      A Dream of the Future
    </h1>
  </header>

  <main>
    <p class="bold">Oh hi,</p>
    <p>Tell me your hopes and dreams:</p>
    <form>
      <input type="text" maxlength="100" placeholder="Dreams!">
      <button type="submit">Submit</button>
    </form>
    <section class="dreams">
      <ul id="dreams">
      </ul>
    </section>
  </main>

  <footer>
    <a href="https://glitch.com">
      Remix this in Glitch
    </a>
  </footer>
```

html

```
<body>
  <header>
    <h1>
      A Dream of the Future
    </h1>
  </header>

  <main>
    <p class="bold">Oh hi,</p>
    <p>Tell me your hopes and dreams:</p>
    <form>
      <input type="text" maxlength="100" placeholder="Your dream" />
      <button type="submit">Submit</button>
    </form>
    <section class="dreams">
      <ul id="dreams">
      </ul>
    </section>
  </main>

  <footer>
    <a href="https://gomix.com">
      Remix this in Gomix
    </a>
  </footer>
```

client side javascript

```
// client-side js
// run by the browser each time your view template is loaded

// by default, you've got jQuery,
// add other scripts at the bottom of index.html

$(function() {
  console.log('hello world :o');

  $.get('/dreams', function(dreams) {
    dreams.forEach(function(dream) {
      $('<li></li>').text(dream).appendTo('ul#dreams');
    });
  });

  $('form').submit(function(event) {
    event.preventDefault();
    dream = $('input').val();
    $.post('/dreams?' + $.param({dream: dream}), function() {
      $('<li></li>').text(dream).appendTo('ul#dreams');
      $('input').val('');
      $('input').focus();
    });
  });
});
```

server side javascript

```
// server.js
// where your node app starts

// init project
var express = require('express');
var app = express();

// we've started you off with Express,
// but feel free to use whatever libs or frameworks you'd like through `package.json`.

// http://expressjs.com/en/starter/static-files.html
app.use(express.static('public'));

// http://expressjs.com/en/starter/basic-routing.html
~ app.get("/", function (request, response) {
  response.sendFile(__dirname + '/views/index.html');
});

~ app.get("/dreams", function (request, response) {
  response.send(dreams);
});

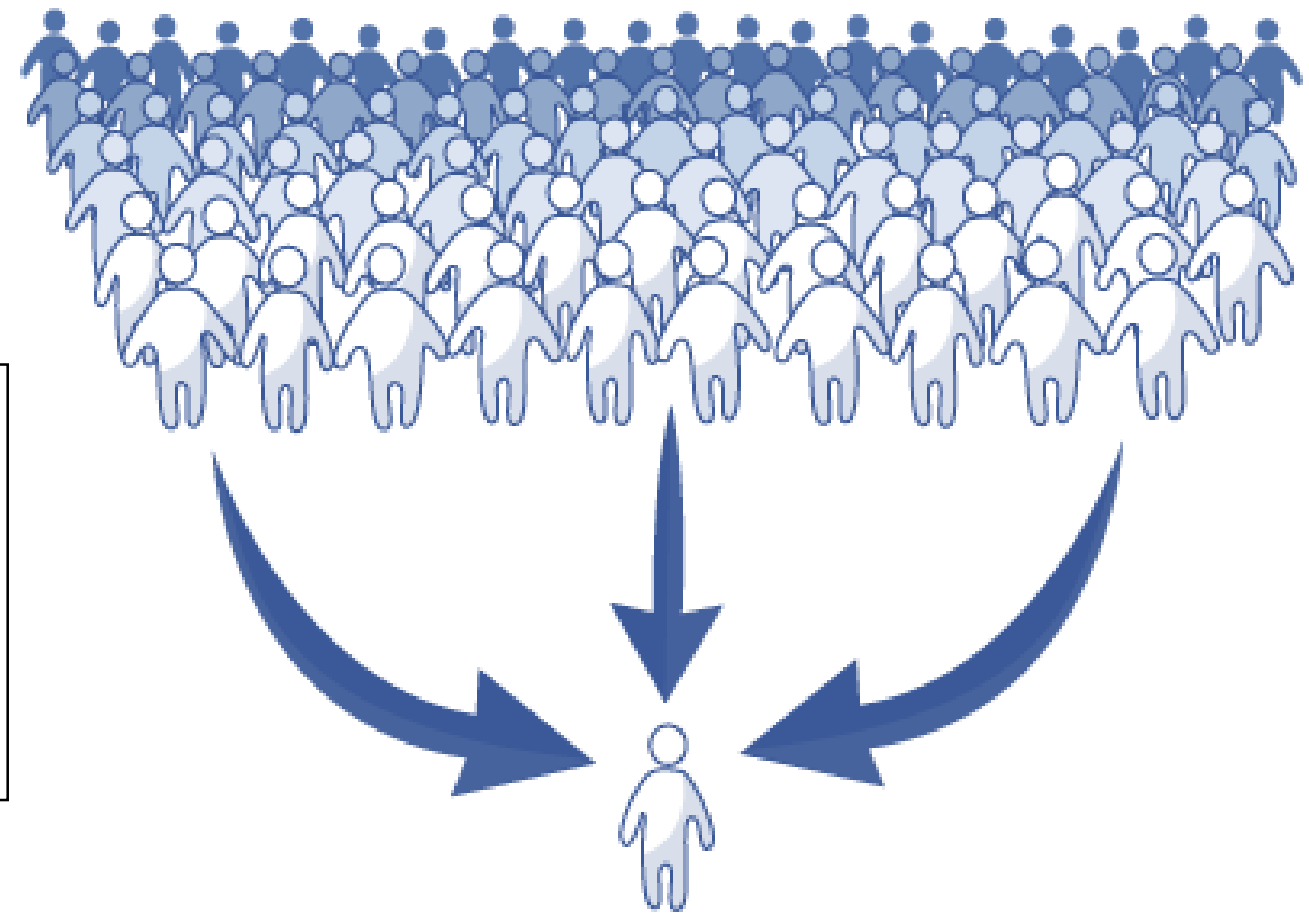
// could also use the POST body instead of query string: http://expressjs.com/en/api.html#req.body
~ app.post("/dreams", function (request, response) {
  dreams.push(request.query.dream);
  response.sendStatus(200);
});

// Simple in-memory store for now
~ var dreams = [
  "Find and count some sheep",
  "Climb a really tall mountain",
  "Wash the dishes"
];

// listen for requests :)
~ var listener = app.listen(process.env.PORT, function () {
  console.log('Your app is listening on port ' + listener.address().port);
});
```


- Client side javascript runs in each users browser

```
$('#form').submit(function(event) {  
  event.preventDefault();  
  dream = $('#input').val();  
  $.post('/dreams?' + $.param({dream: dream}), function() {  
    $('<li></li>').text(dream).appendTo('ul#dreams');  
    $('#input').val('');  
    $('#input').focus();  
  });  
});
```

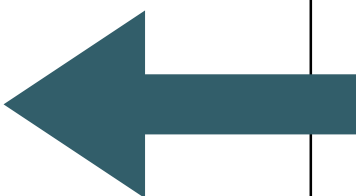


```
// could also use the POST body instead of query string: http://expressjs.com/en/api.html#req.body  
✓ app.post("/dreams", function (request, response) {  
  dreams.push(request.query.dream);  
  response.sendStatus(200);  
});
```

- A node runs the server side javascript. All browsers connected to this node

Skills developed in this Module

- Web App Development 1
 - Basic Javascript knowledge
 - Back end development in Javascript
- Front end javascript development is deferred for a future module



```
// server.js
// where your node app starts

// init project
var express = require('express');
var app = express();

// we've started you off with Express,
// but feel free to use whatever libs or frameworks you'd like through `package.json`.

// http://expressjs.com/en/starter/static-files.html
app.use(express.static('public'));

// http://expressjs.com/en/starter/basic-routing.html
app.get("/", function (request, response) {
  response.sendFile(__dirname + '/views/index.html');
});

app.get("/dreams", function (request, response) {
  response.send(dreams);
});

// could also use the POST body instead of query string: http://expressjs.com/en/api.html#req.body
app.post("/dreams", function (request, response) {
  dreams.push(request.query.dream);
  response.sendStatus(200);
});

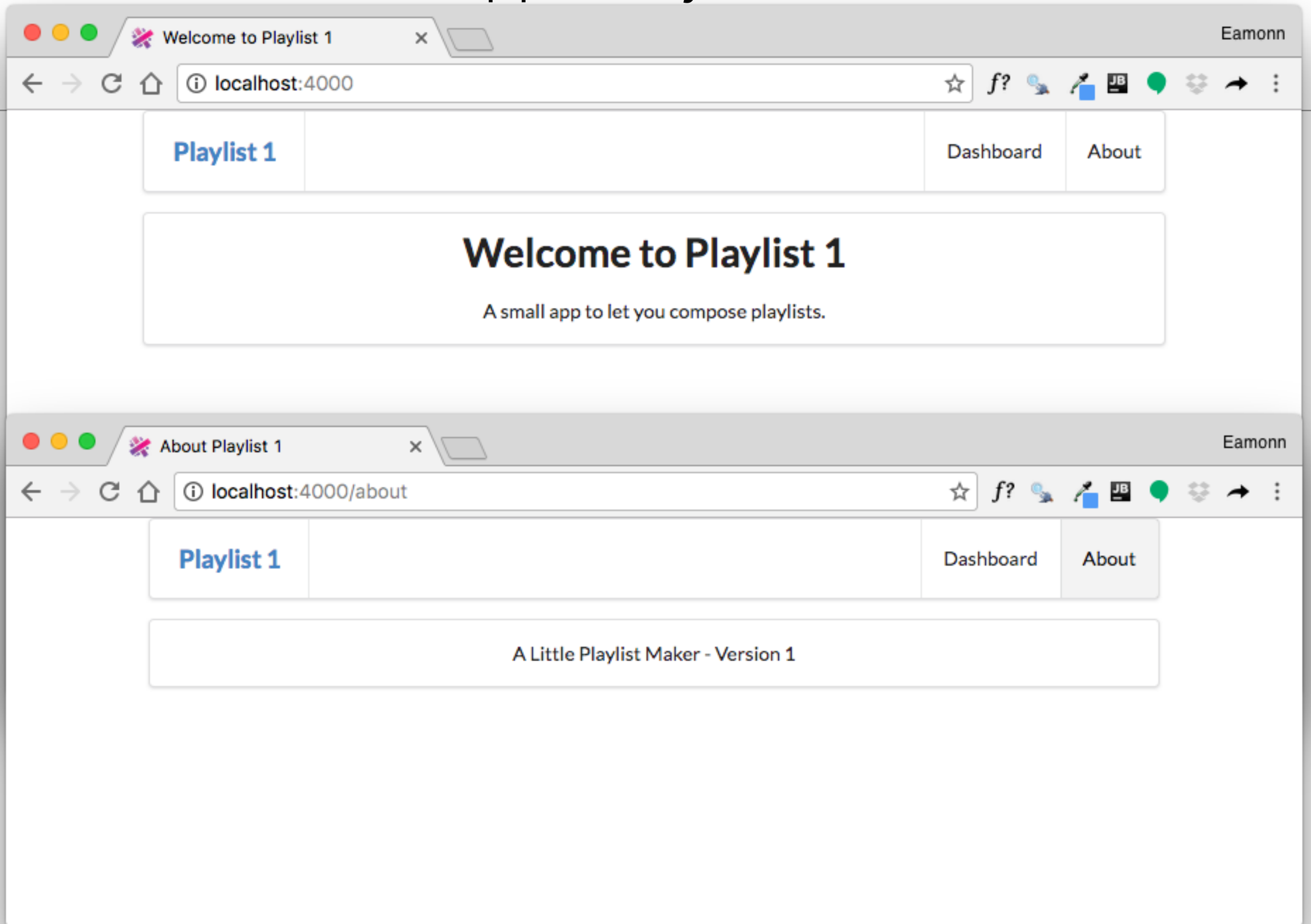
// Simple in-memory store for now
var dreams = [
  "Find and count some sheep",
  "Climb a really tall mountain",
  "Wash the dishes"
];

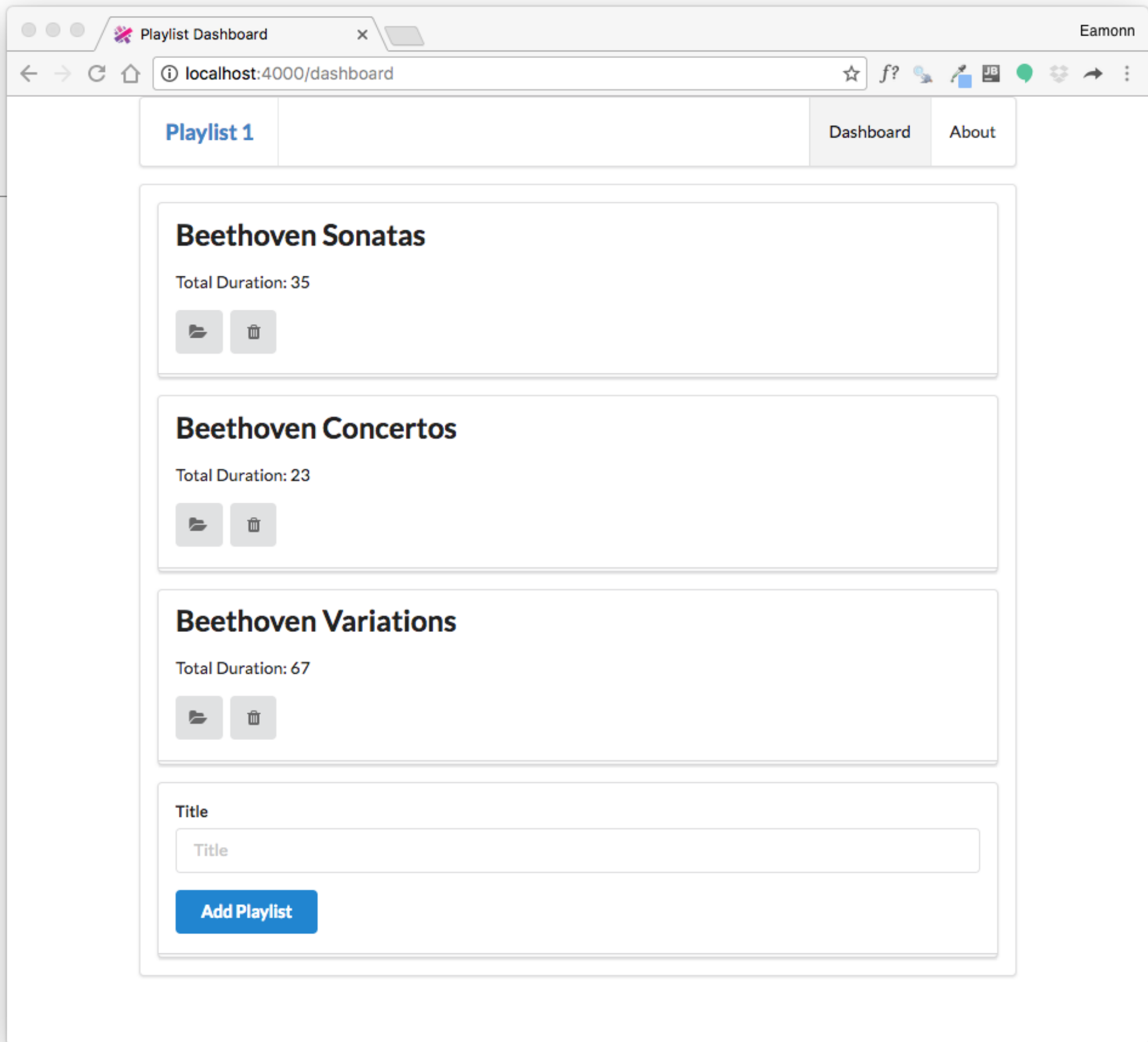
// listen for requests :)
var listener = app.listen(process.env.PORT, function () {
  console.log('Your app is listening on port ' + listener.address().port);
});
```

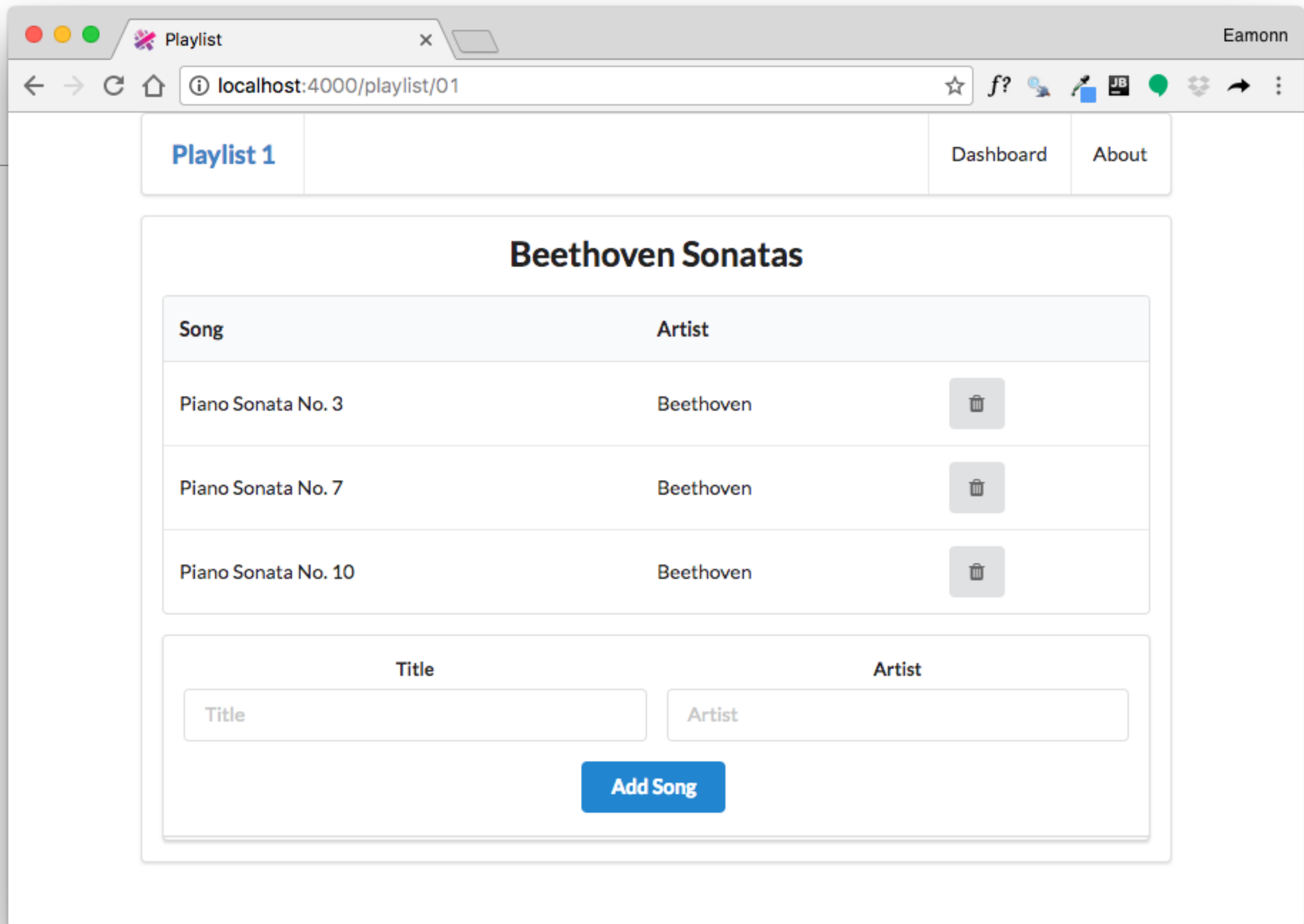
We will learn what all of this means.

- + how to build a fully featured web app including:
 - templating
 - forms to submit information
 - How to store data in models
 - create user accounts, and tie account to a each user

A tour of our first app - Playlist










Playlist 1

Dashboard

About

Beethoven Sonatas

Song	Artist	
Piano Sonata No. 3	Beethoven	
Piano Sonata No. 7	Beethoven	
Piano Sonata No. 10	Beethoven	

Title

Artist

Title

Artist

Add Song

Playlist Labs

- We will do three playlist labs
 - Playlist 1: simple rendering of static playlist
 - Playlist 2: render multiple playlists, ability to delete playlists
 - Playlist 3: ability to create playlists. Store playlists long term.
- These labs will be interleaved with Javascript Introductory labs, which will gradually introduce you to the language