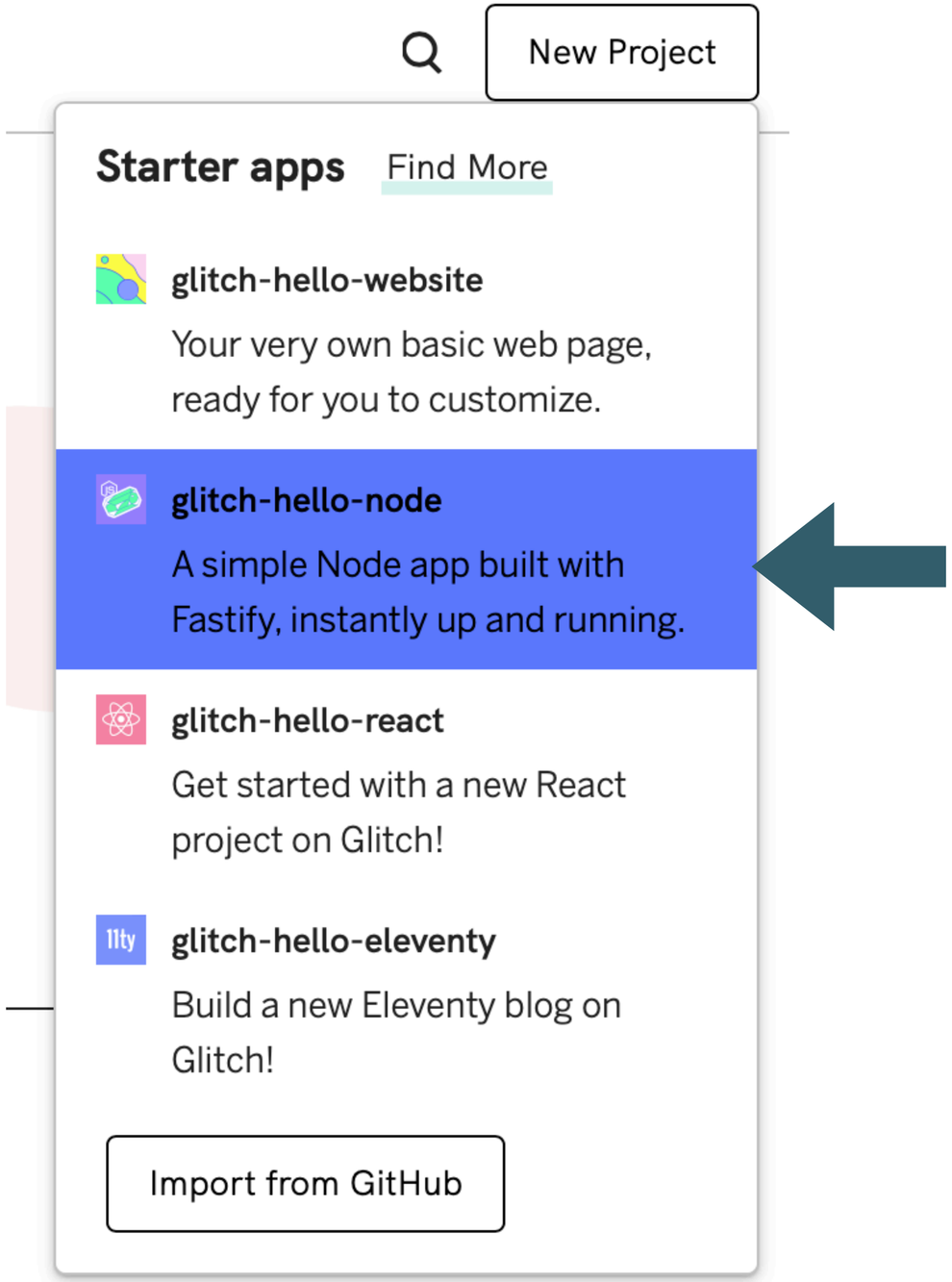
Guided Tour

A look at at the components of a glitch project. Also the types of...

Prerequisite tools on your Workstation

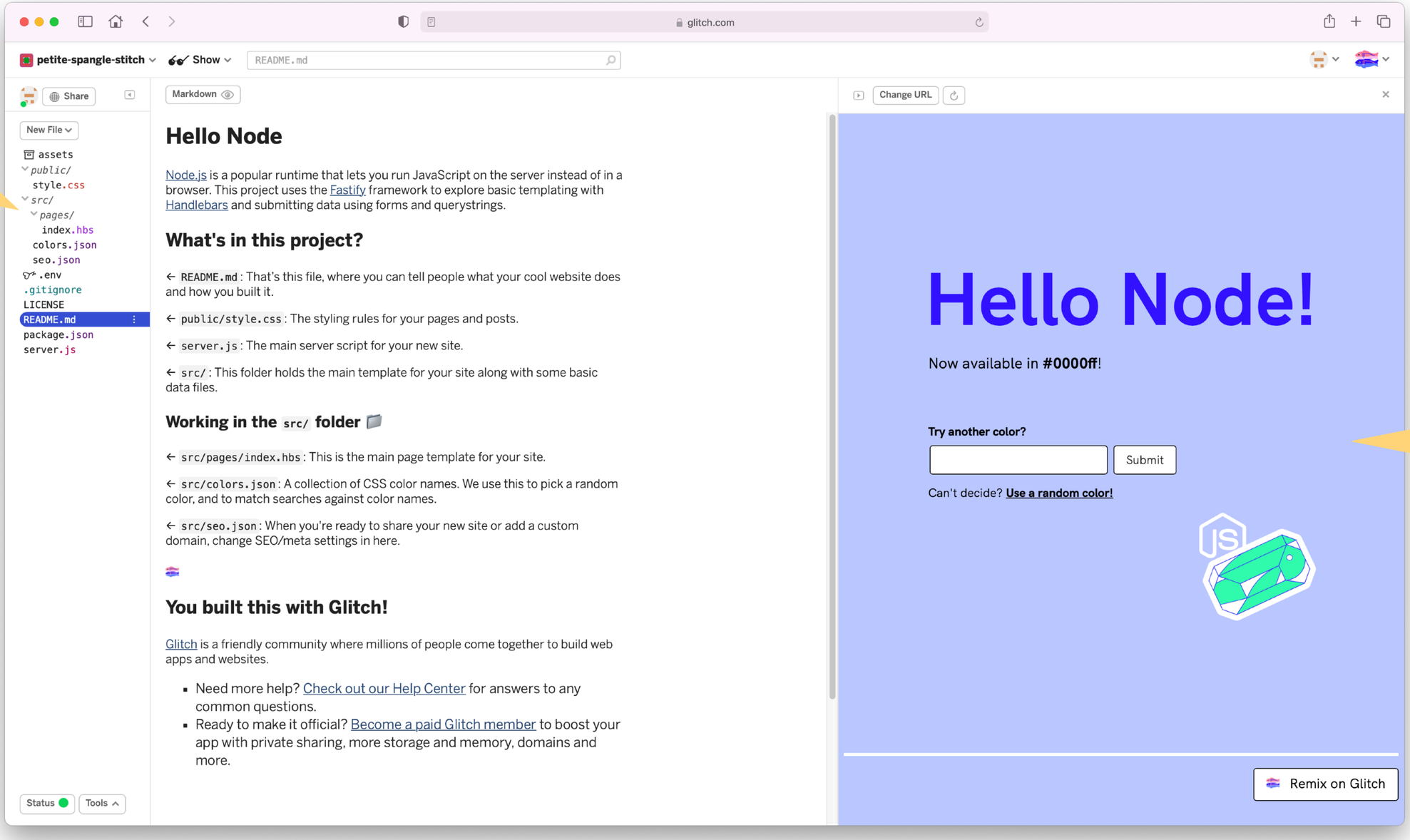
none!

(apart from a browser + a github account)



- The type of project we will be working with

App Source



Running App

Project name (automatically generated)

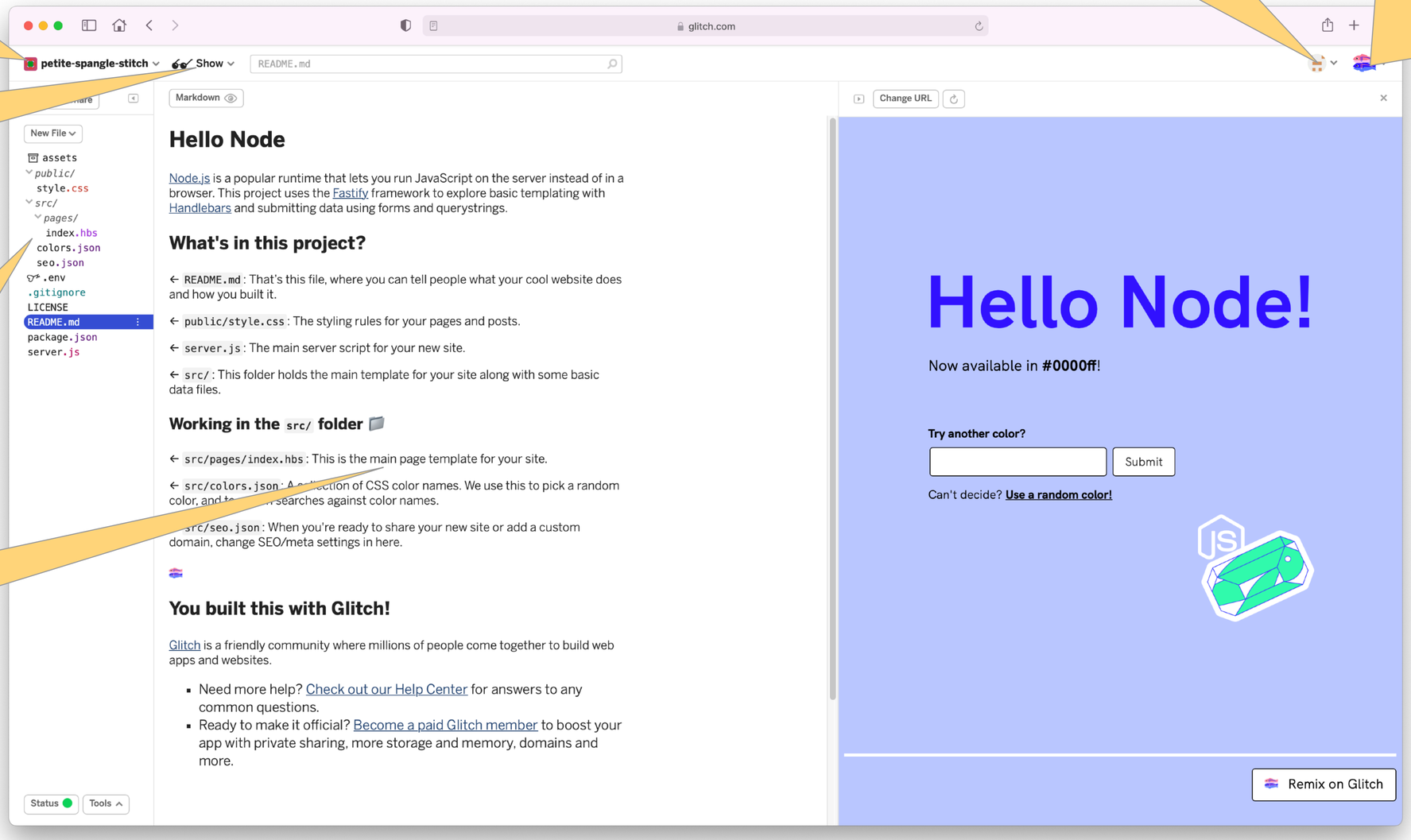
Link to your Profile

Link to Community, resources, options

Link to running app (to share)

Folders/Files in the project

Current File (editable)



glitch.com

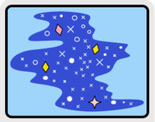
petite-spangle-stitch | Show | README.md

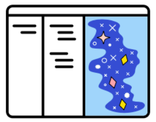
Share

New File

- assets
- public/
  - style.css
- src/
  - pages/
    - index.hbs
  - colors.json
  - seo.json
- .env
- .gitignore
- LICENSE
- README.md
- package.json
- server.js

Status | Tools


In a New Window


Next to The Code

ts you run JavaScript on the server instea
browser. This project uses the Fastify framework to explore basic templating
Handlebars and submitting data using forms and querystrings.

What's in this project?

← `README.md` : That's this file, where you can tell people what your cool webs
and how you built it.

← `public/style.css` : The styling rules for your pages and posts.

← `server.js` : The main server script for your new site.

← `src/` : This folder holds the main template for your site along with some b
data files.

Working in the `src/` folder

← `src/pages/index.hbs` : This is the main page template for your site.

← `src/colors.json` : A collection of CSS color names. We use this to pick a r
color, and to match searches against color names.

← `src/seo.json` : When you're ready to share your new site or add a custom
domain, change SEO/meta settings in here.

You built this with Glitch!

Glitch is a friendly community where millions of people come together to bui
apps and websites.

- Need more help? Check out our Help Center for answers to any common
questions.
- Ready to make it official? Become a paid Glitch member to boost your app with
private sharing, more storage and memory, domains and more.

petite-spangle-stitch.glitch.me

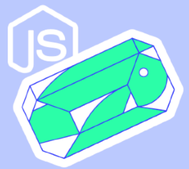
Hello Node!

Learn about communication with a server by talking about color.

What's your favorite color?

Submit

Can't decide? **Use a random color!**



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

Project Structure

- Glitch projects not just web sites!
- They are fully featured web apps - with full server-side resources

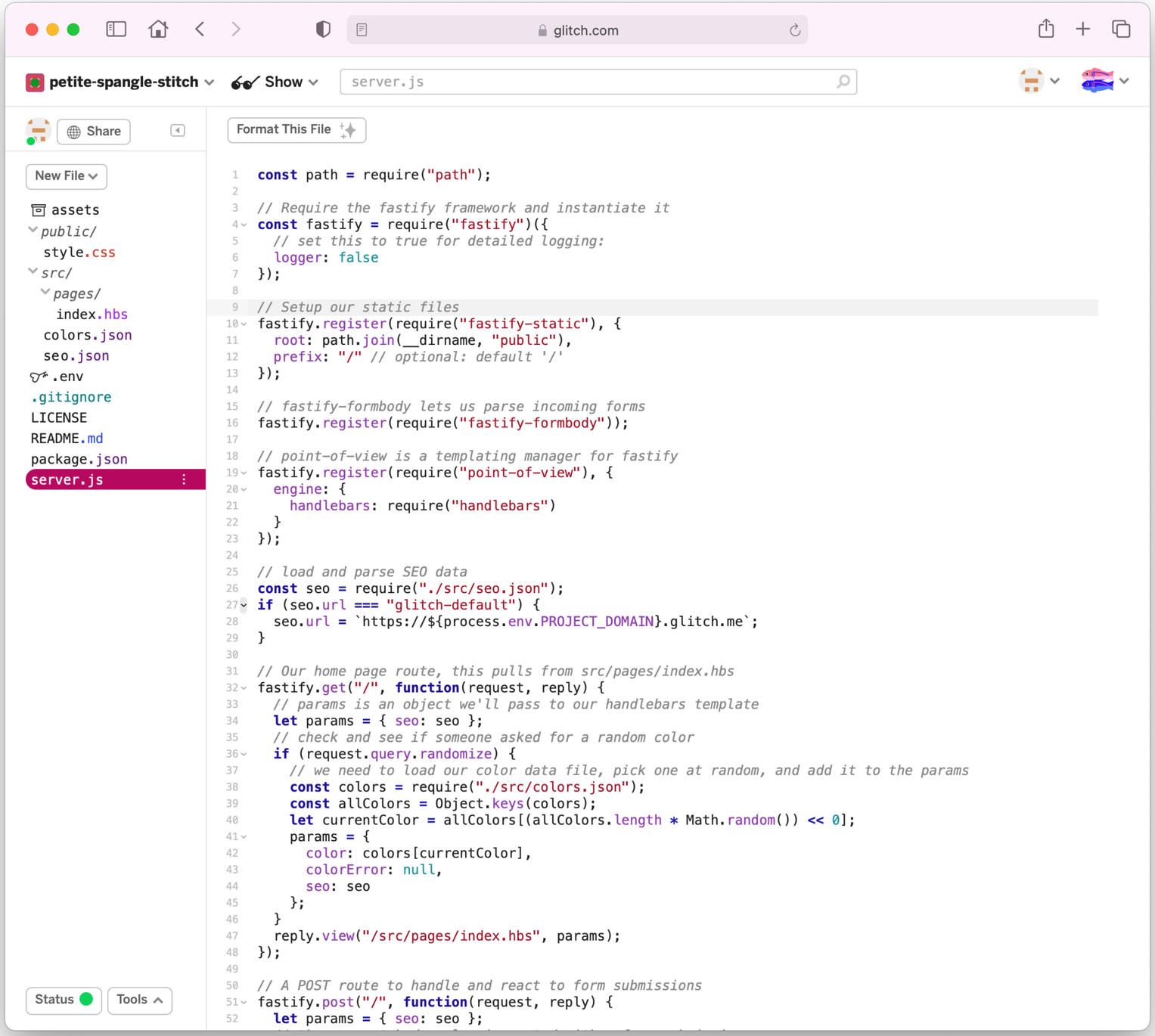
Front End

- Comparable to a static web site:
  - 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)

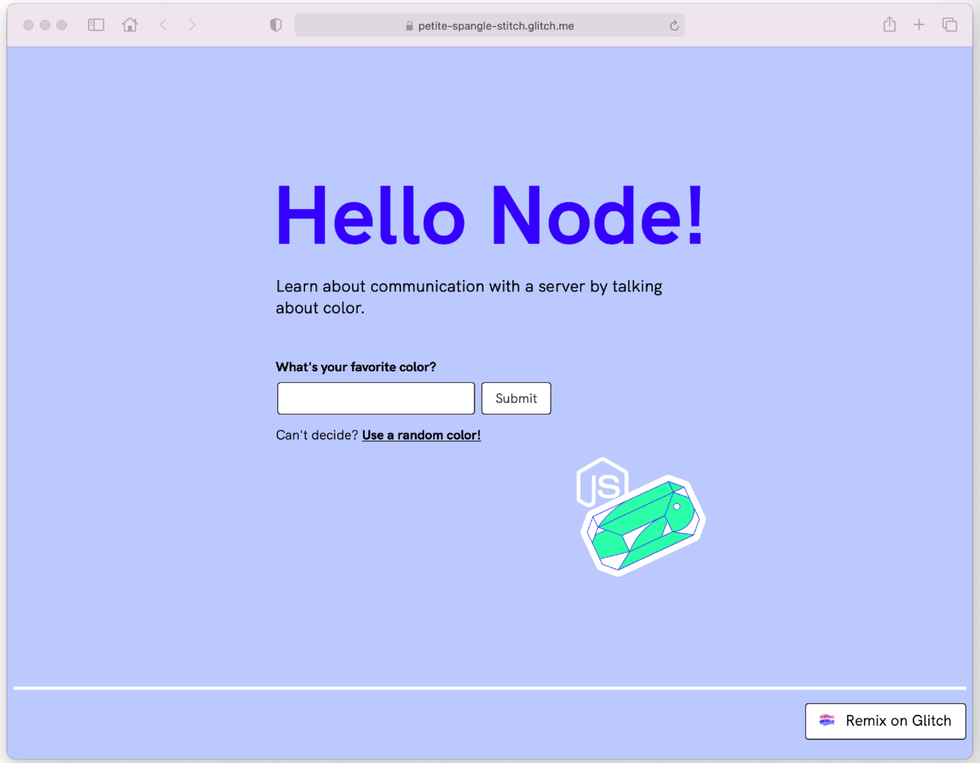
Back end

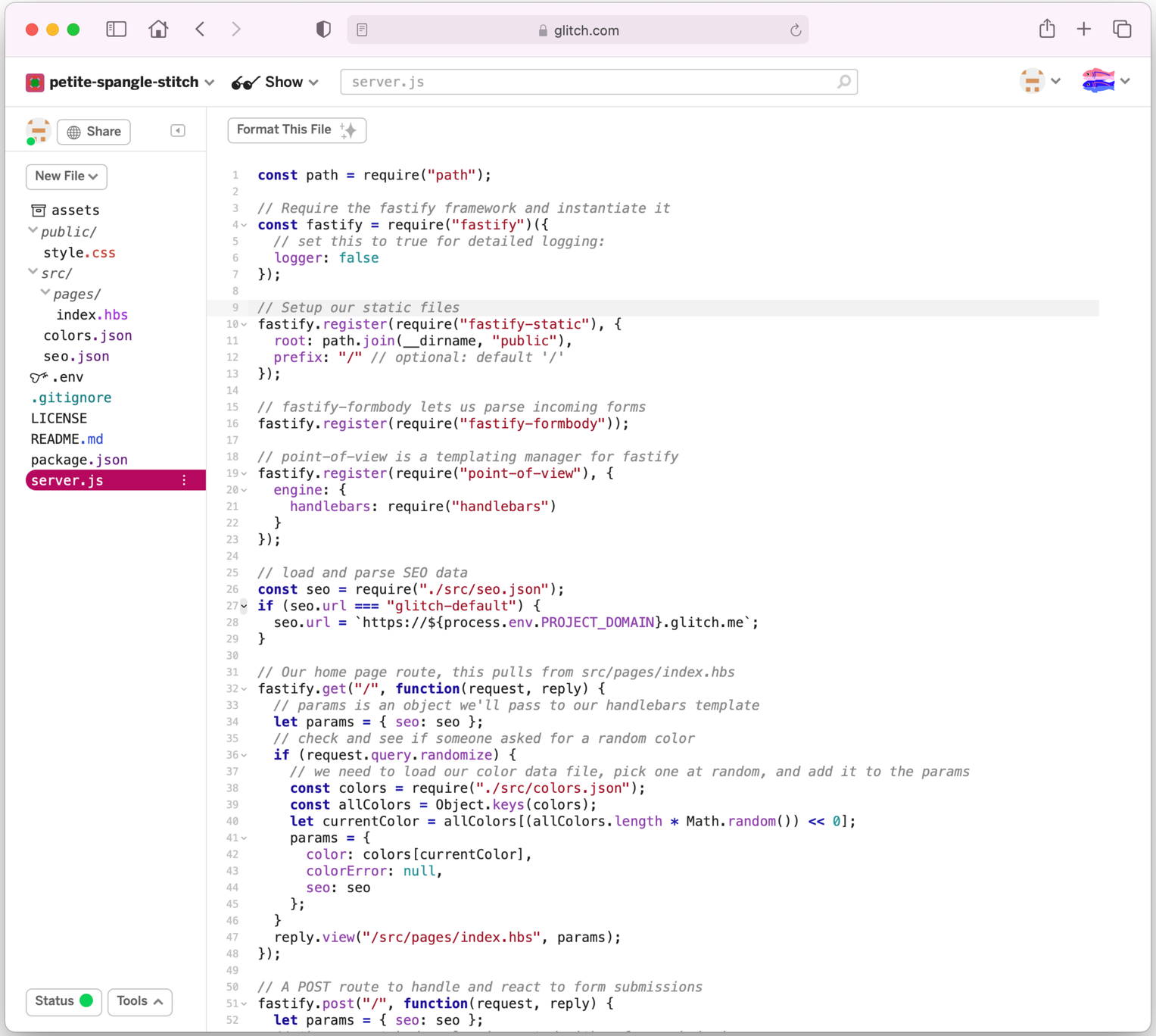
- An application - written in javascript - and hosted in the cloud.
- Application built in Javascript using a technology called node.js

The Starter App

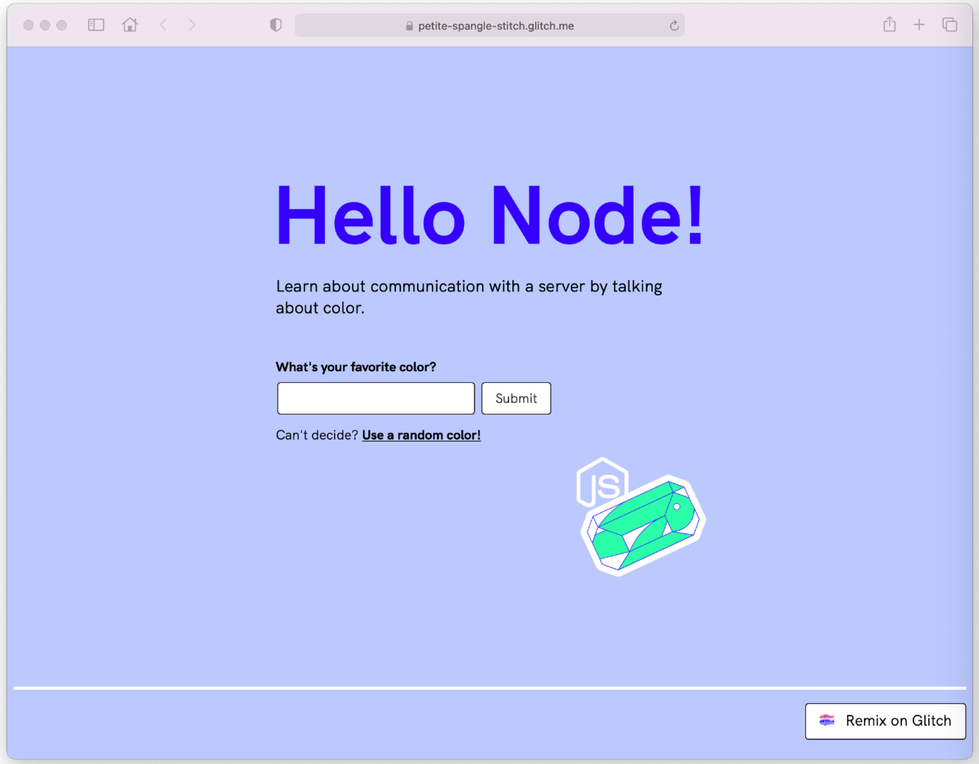


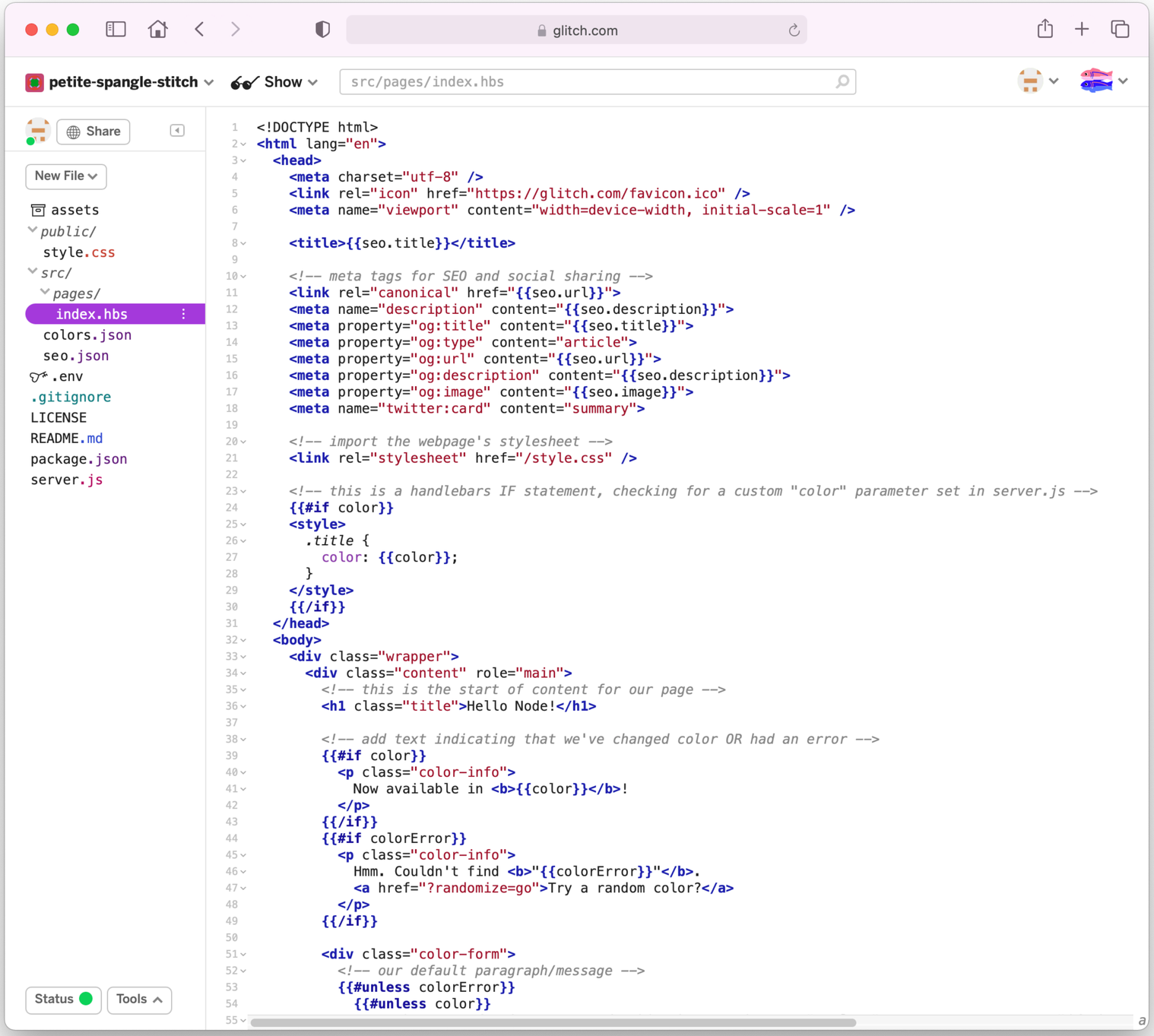
The Starter App





The Starter App





html

```
<div class="color-form">
  <!-- our default paragraph/message -->
  {{#unless colorError}}
    {{#unless color}}
    <!-- There are fancier ways to do this, but nesting two "#unless" statements sets up a "if thes
    <p>
      Learn about communication with a server by talking about color.
    </p>
    {{/unless}}
  {{/unless}}

  <!-- the "what's your favorite color?" form sends a POST to the server, check server.js to see ho
  <form class="color-search" method="post">
      <label for="color">
        <!-- we use the handlebars "#if" statement so the form can adapt to its situation -->
        {{#if color}}
          Try another color?<br>
        {{else}}
          What's your favorite color?<br>
        {{/if}}
        <input id="color" name="color" required="required" type="text">
      </label>

    <button type="submit">Submit</button>
  </form>

  <!-- this randomize link communicates with the server (server.js) using a querystring and GET req
  Can't decide? <a href="?randomize=go">Use a random color!</a>
```

Server side javascript

```
// Our home page route, this pulls from src/pages/index.hbs
fastify.get("/", function(request, reply) {
  // params is an object we'll pass to our handlebars template
  let params = { seo: seo };
  // check and see if someone asked for a random color
  if (request.query.randomize) {
    // we need to load our color data file, pick one at random, and add it to the params
    const colors = require("./src/colors.json");
    const allColors = Object.keys(colors);
    let currentColor = allColors[(allColors.length * Math.random()) << 0];
    params = {
      color: colors[currentColor],
      colorError: null,
      seo: seo
    };
  }
  reply.view("/src/pages/index.hbs", params);
});

// A POST route to handle and react to form submissions
fastify.post("/", function(request, reply) {
  let params = { seo: seo };
  // the request.body.color is posted with a form submission
  let color = request.body.color;
  // if it's not empty, let's try to find the color
  if (color) {
    // load our color data file
    const colors = require("./src/colors.json");
    // take our form submission, remove whitespace, and convert to lowercase
    color = color.toLowerCase().replace(/\s/g, "");
    // now we see if that color is a key in our colors object
    if (colors[color]) {
      // found one!
      params = {
        color: colors[color],
        colorError: null,
        seo: seo
      };
    } else {
      // try again.
      params = {
        colorError: request.body.color,
        seo: seo
      };
    }
  }
  reply.view("/src/pages/index.hbs", params);
});
```

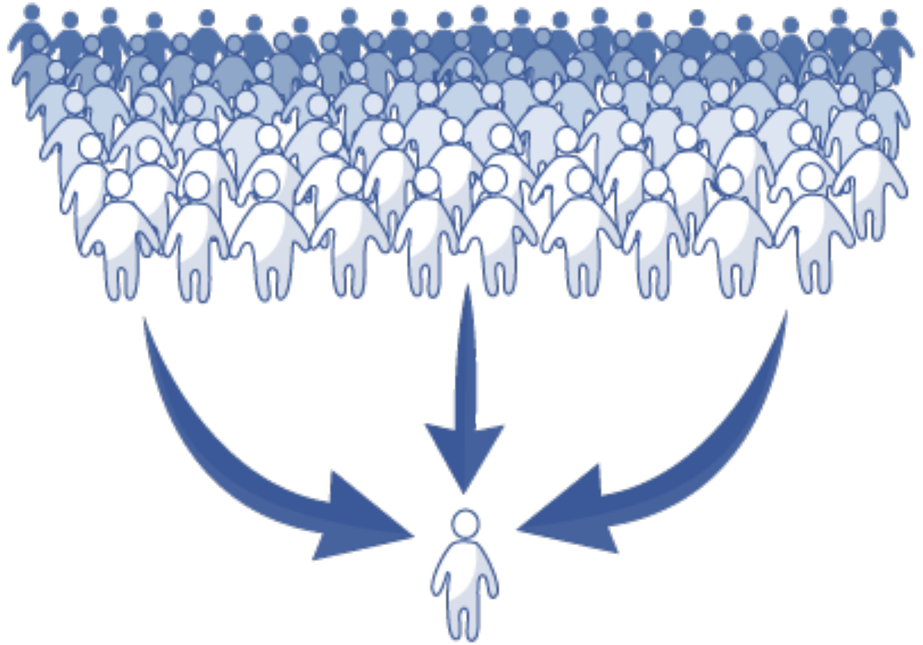
Client side html runs in each users browser

```
<div class="color-form">
  <!-- our default paragraph/message -->
  {{#unless colorError}}
    {{#unless color}}
    <!-- There are fancier ways to do this, but nesting two "#unless" statements sets up a "if these
    <p>
      Learn about communication with a server by talking about color.
    </p>
    {{/unless}}
  {{/unless}}

  <!-- the "what's your favorite color?" form sends a POST to the server, check server.js to see ho
  <form class="color-search" method="post">
      <label for="color">
        <!-- we use the handlebars "#if" statement so the form can adapt to its situation -->
        {{#if color}}
          Try another color?<br>
        {{else}}
          What's your favorite color?<br>
        {{/if}}
        <input id="color" name="color" required="required" type="text">
      </label>

    <button type="submit">Submit</button>
  </form>

  <!-- this randomize link communicates with the server (server.js) using a querystring and GET req
  Can't decide? <a href="?randomize=go">Use a random color!</a>
```



Server side run on a cloud server. All browsers connected to this node

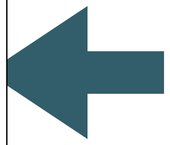
```
// A POST route to handle and react to form submissions
fastify.post("/", function(request, reply) {
  let params = { seo: seo };
  // the request.body.color is posted with a form submission
  let color = request.body.color;
  // if it's not empty, let's try to find the color
  if (color) {
    // load our color data file
    const colors = require("./src/colors.json");
    // take our form submission, remove whitespace, and convert to lowercase
    color = color.toLowerCase().replace(/\s/g, "");
    // now we see if that color is a key in our colors object
    if (colors[color]) {
      // found one!
      params = {
        color: colors[color],
        colorError: null,
        seo: seo
      };
    } else {
      // try again.
      params = {
        colorError: request.body.color,
        seo: seo
      };
    }
```

Skills for this Course

- Assumptions:
  - Foundation Knowledge in HTML + CSS
  - Working knowledge of Semantic UI CSS Framework
  - Familiarity with Play Framework + Todo & Playlist applications
- Major focus of this course:
  - Back end Javascript Programming
  - Node.js Web Application Development
- Glitch is the platform + WebStorm (sibling of IDEA) as alternative
- Front end javascript development will **not** be covered.

```
// Our home page route, this pulls from src/pages/index.hbs
fastify.get("/", function(request, reply) {
  // params is an object we'll pass to our handlebars template
  let params = { seo: seo };
  // check and see if someone asked for a random color
  if (request.query.randomize) {
    // we need to load our color data file, pick one at random, and add it to the params
    const colors = require("./src/colors.json");
    const allColors = Object.keys(colors);
    let currentColor = allColors[(allColors.length * Math.random()) << 0];
    params = {
      color: colors[currentColor],
      colorError: null,
      seo: seo
    };
  }
  reply.view("/src/pages/index.hbs", params);
});

// A POST route to handle and react to form submissions
fastify.post("/", function(request, reply) {
  let params = { seo: seo };
  // the request.body.color is posted with a form submission
  let color = request.body.color;
  // if it's not empty, let's try to find the color
  if (color) {
    // load our color data file
    const colors = require("./src/colors.json");
    // take our form submission, remove whitespace, and convert to lowercase
    color = color.toLowerCase().replace(/\s/g, "");
    // now we see if that color is a key in our colors object
    if (colors[color]) {
      // found one!
      params = {
        color: colors[color],
        colorError: null,
        seo: seo
      };
    } else {
      // try again.
      params = {
        colorError: request.body.color,
        seo: seo
      };
    }
  }
  reply.view("/src/pages/index.hbs", params);
});
```



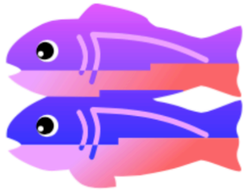
- We will learn what all of this means.
- + how to build a fully featured web app including:
  - templating
  - forms to submit information
  - How store information in models
  - create user accounts, and tie account to a each user

All of this requires beginner/intermediate level Javascript skills

Preview of Course Labs

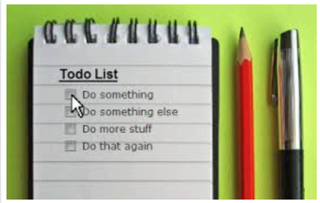
- Introductory
- Javascript
- Playlist
- Todolist

Lab-00a Glitch Intro



Create, modify and view your first Glitch project.

Lab-00b Todo 1



Create a Todo application in Glitch

Lab-00c Todo 2

Incorporate IDs into the todo model

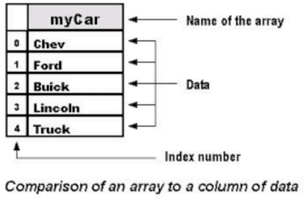
Lab-00d Todo 3

Incorporate Sessions / User accounts into Todo

Lab-01 JS Intro

Background & Tools, Variables & Boolean Logic

Lab-02 JS Arrays



Comparison of an array to a column of data

Array Basics, Array Methods & Iteration

Lab-03a Playlist 1

Import and run a new starter project. Extend this project to include multiple..

Lab-03b WebStorm

Using WebStorm instead of Glitch for Web App development

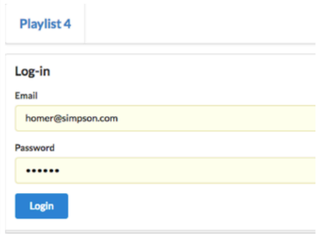
Lab-04 Playlist 2

Refactor the dashboard controller to show summary on of the playlists + link to..

Lab-05 Playlist 3

Enable Songs and Playlists to be added via simple forms.

Lab-06a Playlist 4



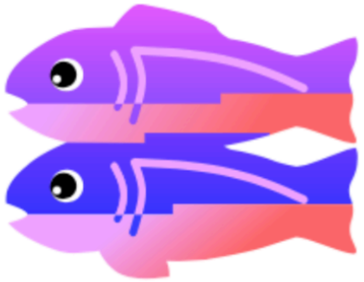
Introduce Sessions onto the Playlist application, enabling user accounts a...

Lab-06b Playlist 5

Implement a feature to update an existing Song

Introductory

Lab-00a Glitch Intro



Create, modify and view your first Glitch project.

Lab-00b Todo 1

Create a Todo application in Glitch

Lab-00c Todo 2

Incorporate IDs into the todo model

Lab-00d Todo 3

Incorporate Sessions / User accounts into Todo

Exploring the Look & Feel of Glitch

Advance preview of course. Recreate the play todo list applications in Javascript Glitch.
Feel free to Skip these three labs (for the moment)!

Javascript

Accelerated introducing to Javascript

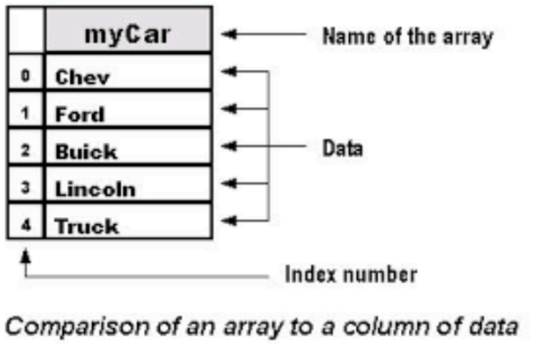
Explore a subset of the language suitable for our purposes

NOT using glitch for these labs - just a web browser

Lab-01 JS Intro

Background & Tools, Variables & Boolean Logic

Lab-02 JS Arrays



Comparison of an array to a column of data

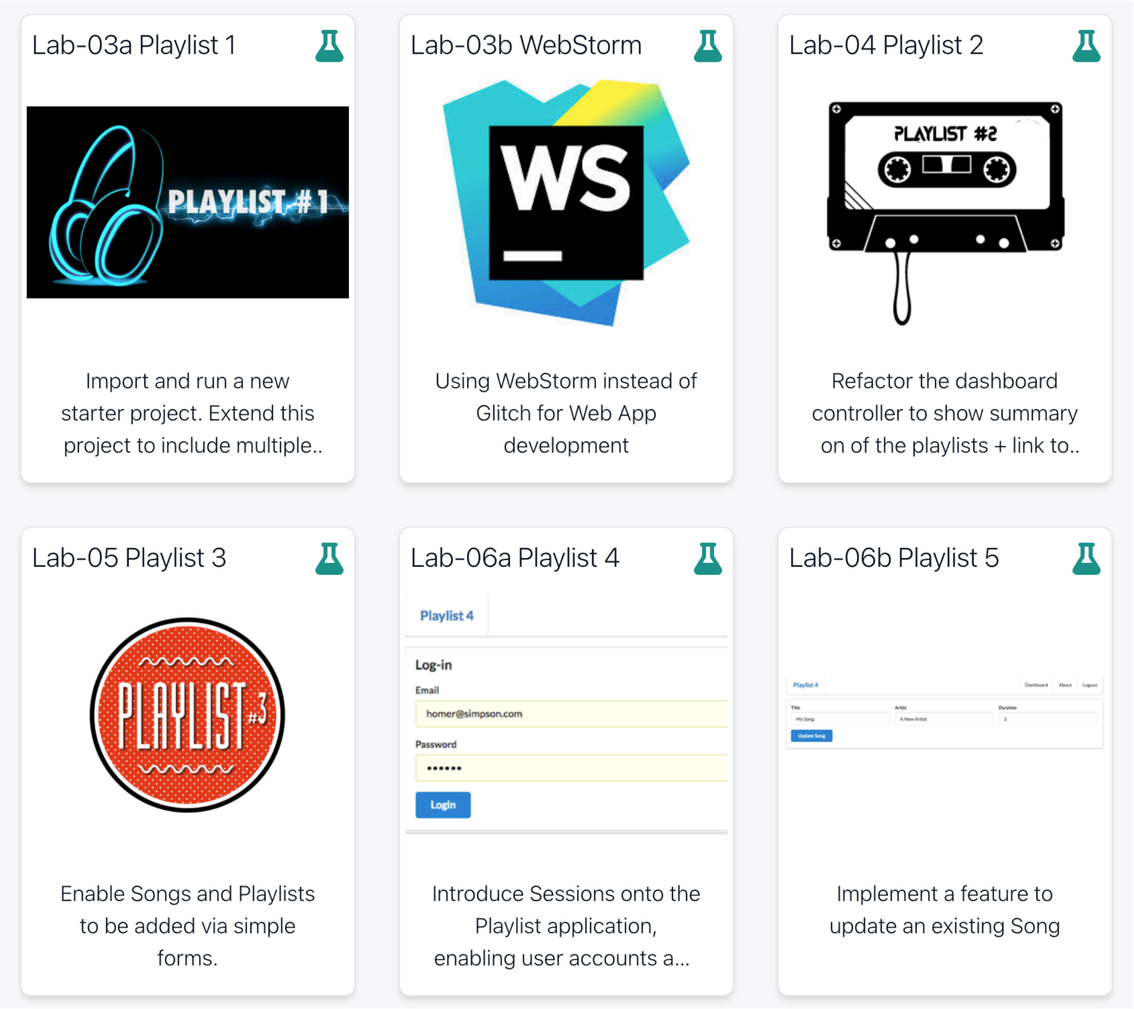
Array Basics, Array Methods & Iteration

Playlist

Rebuild the Playlist Applications

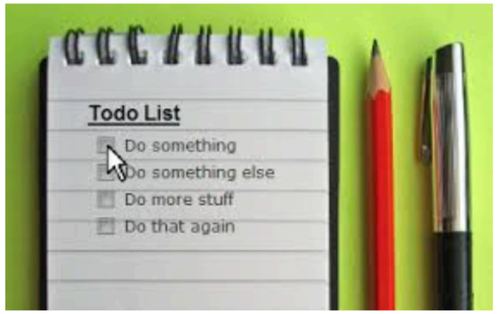
Identical features from Play/Java

Implemented in Javascript/ Express



Todolist

Lab-00b Todo 1



Create a Todo application in Glitch

Lab-00c Todo 2

Incorporate IDs into the todo model

Lab-00d Todo 3

Incorporate Sessions / User accounts into Todo

Do/Redo these labs as a revision of the core concepts

Guided Tour

A look at at the components of a glitch project. Also the types of...