Building a Backend API

with Netlify Serverless Functions, Express and PostgreSQL

Building a Scalable Backend API with Serverless Functions

Building a backend API with Netlify functions (on top of AWS Lambda functions), Node.js, Express, and PostgreSQL can be a powerful and efficient way to create scalable and cost-effective applications.

With the right tools and techniques, developers can create robust and flexible APIs that can be easily deployed and managed on the cloud.

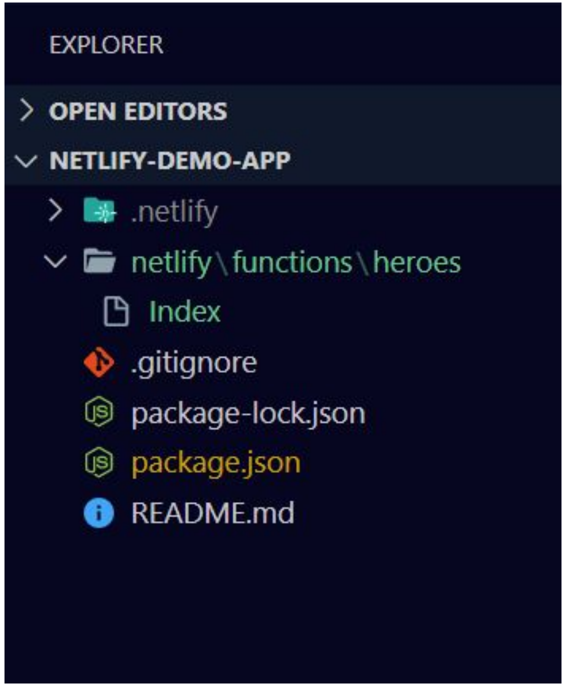
Basic Project Structure

The project structure consists of several directories and files. The Netlify CLI creates a ".netlify" directory automatically. This directory contains configuration files for Netlify deployment.

The "netlify" directory contains the serverless functions directory "functions". Inside "heroes", there is a subdirectory "heroes" that contains the "index.js" file which represents the backend API.

The "node_modules" directory contains all the dependencies needed for the project, which are installed using the "package.json" and "package-lock.json" files.

The ".gitignore" file is used to exclude certain files and directories from being tracked by git, while the "README.md" file is used to provide documentation and instructions for the project.



```
EXPLORER
> OPEN EDITORS
v NETLIFY-DEMO-APP
  > .netlify
  v netlify\functions\heroes
      Index
    .gitignore
    package-lock.json
    package.json
    README.md
```

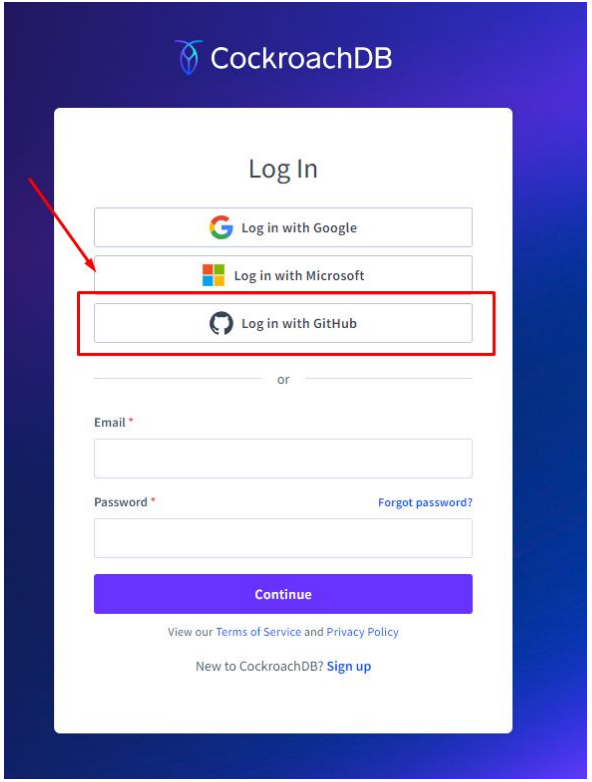
CockroachDB

The world's most evolved cloud SQL database — giving all of your apps effortless scale, bulletproof resilience and low latency performance for users anywhere.

Deploy a Netlify App Built on CockroachDB

Step 1

Log in to your CockroachDB Cloud account.


CockroachDB

Log In

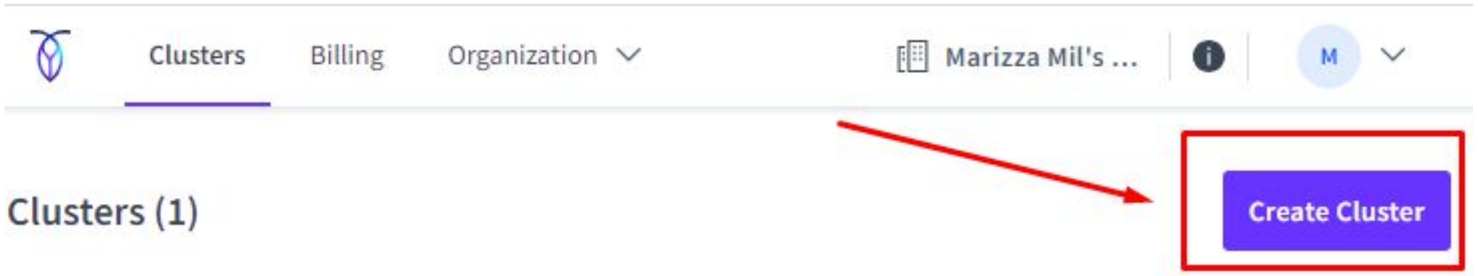
View our Terms of Service and Privacy Policy

New to CockroachDB? Sign up

Deploy a Netlify App Built on CockroachDB

Step 2

On the **Clusters** page, click **Create Cluster**.



Deploy a Netlify App Built on CockroachDB

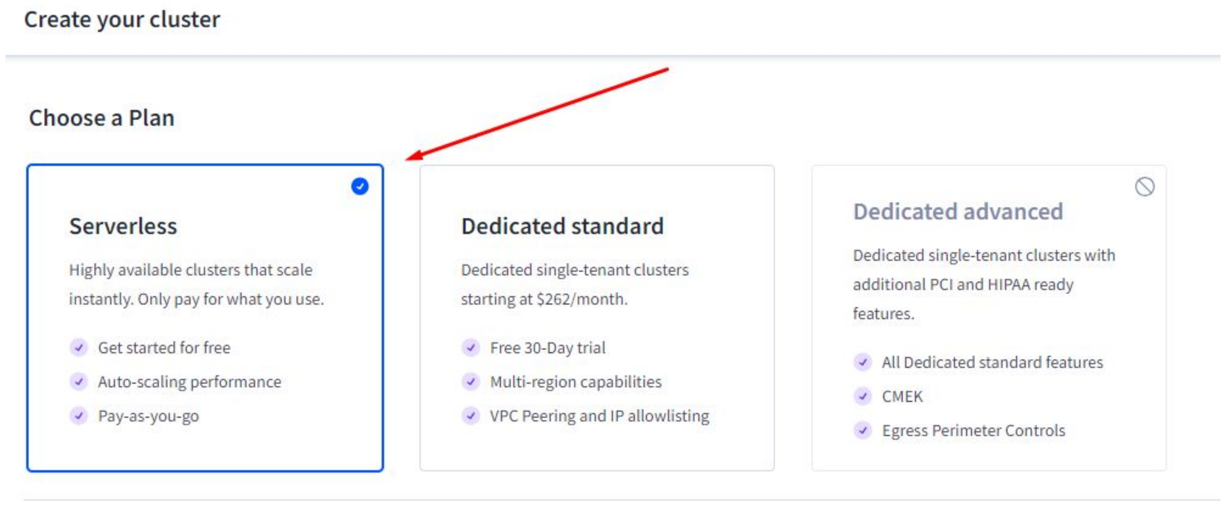
Step 3

On the **Create your cluster** page, select **Serverless**.

Step 4

Click **Create cluster**.

Your cluster will be created in a few seconds and the **Create SQL user** dialog will display.



Create your cluster

Choose a Plan

Serverless ✓

Highly available clusters that scale instantly. Only pay for what you use.

- ✓ Get started for free
- ✓ Auto-scaling performance
- ✓ Pay-as-you-go

Dedicated standard

Dedicated single-tenant clusters starting at $262/month.

- ✓ Free 30-Day trial
- ✓ Multi-region capabilities
- ✓ VPC Peering and IP allowlisting

Dedicated advanced

Dedicated single-tenant clusters with additional PCI and HIPAA ready features.

- ✓ All Dedicated standard features
- ✓ CMEK
- ✓ Egress Perimeter Controls

Deploy a Netlify App Built on CockroachDB

Click **Create cluster**.

Your cluster will be created in a few seconds and the **Create SQL user** dialog will display.

Creating a Simple Express App for Netlify Serverless Function

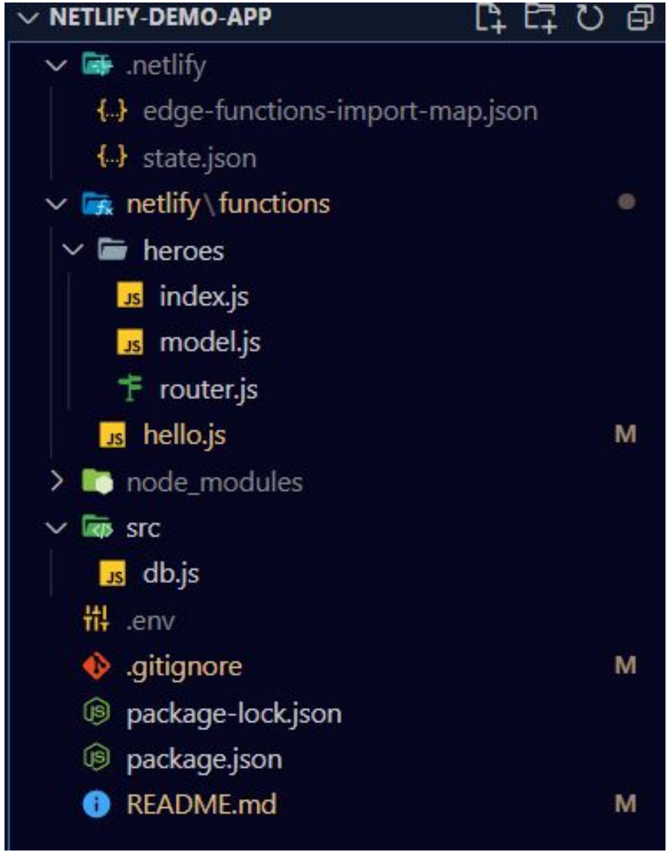
This code sets up a simple Express app with two route handlers and exports it as a serverless function that can be deployed on Netlify. When a request is made to one of the defined routes, the app will respond with the appropriate message or data.

To use the code, you need to have the dependencies "express", "serverless-http", "pg" and "dotenv" installed. To install these dependencies, you can run the following command in your terminal while in the root directory of your project:

```
npm install express serverless-http pg dotenv
```

Project Structure

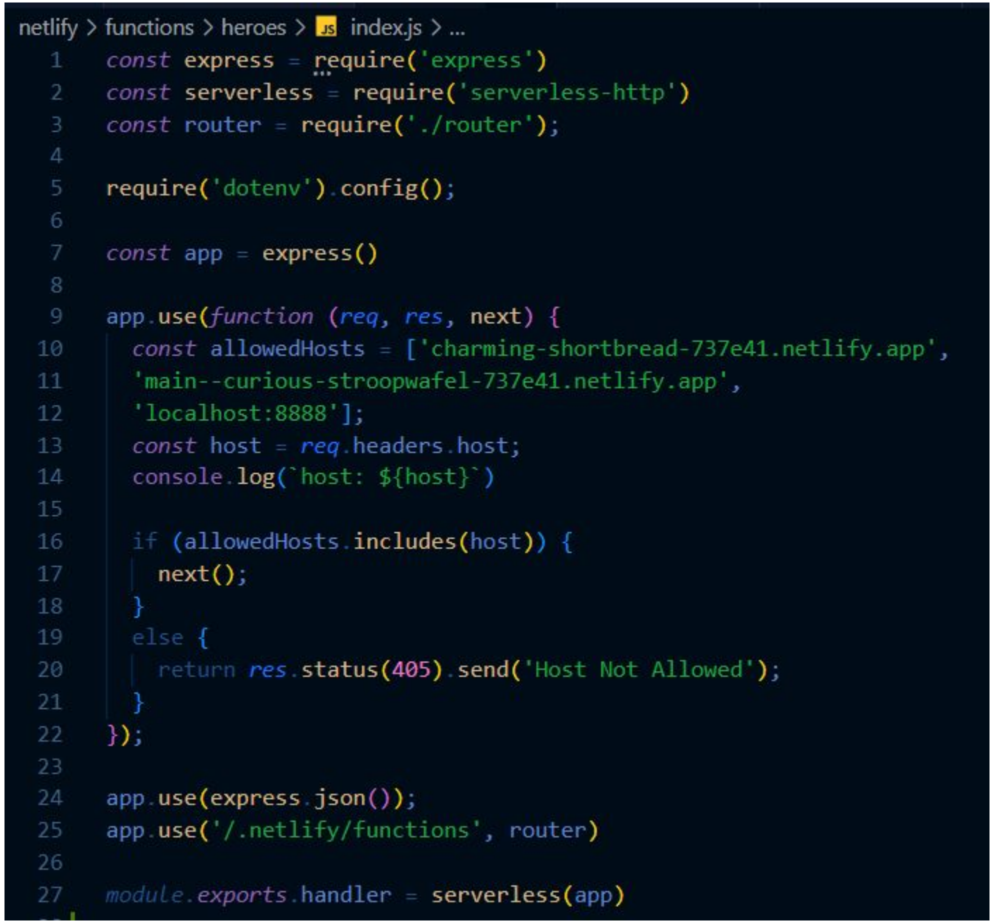
- .netlify: Contains files and configurations specific to deploying the app on Netlify.
    i. netlify/functions/heroes/index.js: Entry point for the Netlify serverless function handling CRUD operations for heroes.
    ii. netlify/functions/heroes/model.js: Defines the model responsible for database interactions.
    iii. netlify/functions/heroes/router.js: Defines the API routes for CRUD operations.
- node_modules: Directory where Node.js dependencies are installed.
- src: Main source code directory for the application.
    i. db.js: Defines the connection and initialization of the CockroachDB database using credentials from the .env file.
- .env: Configuration file for storing sensitive information like CockroachDB credentials.
- .gitignore: Specifies files and directories that should be ignored by version control (e.g., node_modules, .env, etc.).
- package-lock.json: Automatically generated file specifying the exact versions of installed dependencies.
- package.json: Manifest file defining the project's metadata, dependencies, scripts, and more.
- README.md: Documentation providing an overview of the project, installation instructions, and usage guidelines.


```
NETLIFY-DEMO-APP
  .netlify
    edge-functions-import-map.json
    state.json
  netlify\functions
    heroes
      index.js
      model.js
      router.js
    hello.js                          M
  node_modules
  src
    db.js
  .env
  .gitignore                          M
  package-lock.json
  package.json
  README.md                           M
```

Full Code for New Project Structure

`./netlify/function/heroes/index.js`

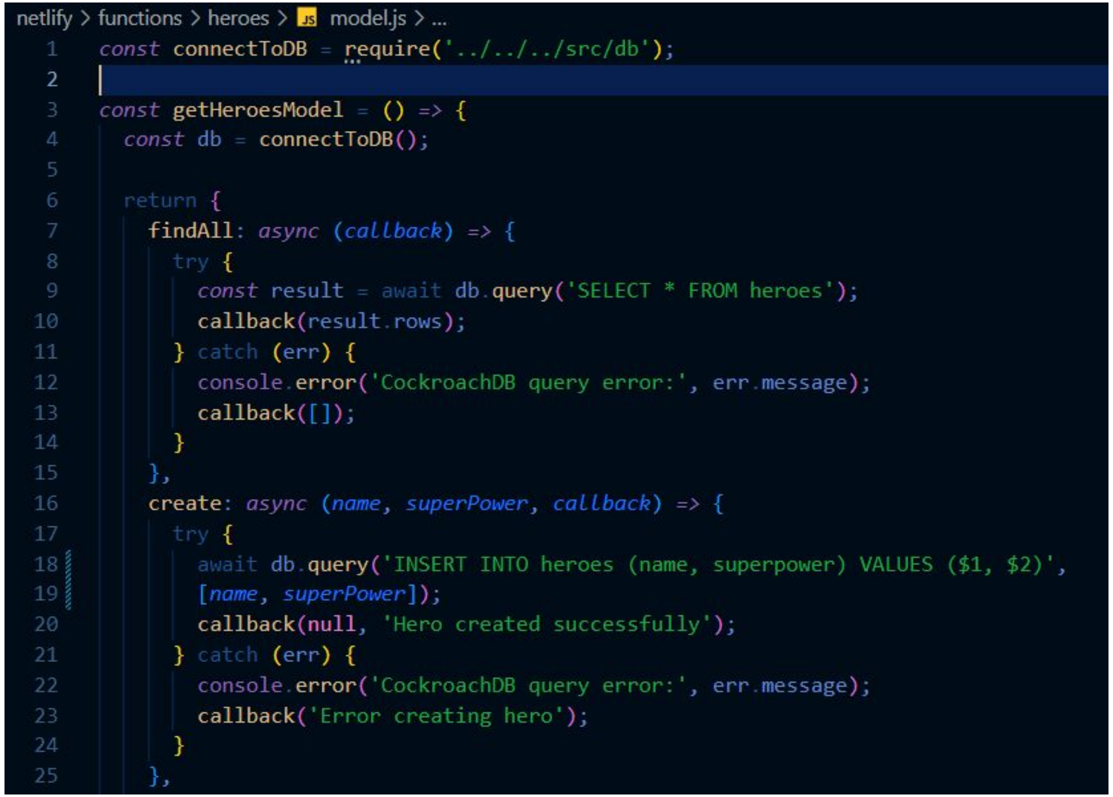
This file is the entry point for the Netlify function.



```javascript
const express = require('express')
const serverless = require('serverless-http')
const router = require('./router');

require('dotenv').config();

const app = express()

app.use(function (req, res, next) {
  const allowedHosts = ['charming-shortbread-737e41.netlify.app',
  'main--curious-stroopwafel-737e41.netlify.app',
  'localhost:8888'];
  const host = req.headers.host;
  console.log(`host: ${host}`)

  if (allowedHosts.includes(host)) {
    next();
  }
  else {
    return res.status(405).send('Host Not Allowed');
  }
});

app.use(express.json());
app.use('/.netlify/functions', router)

module.exports.handler = serverless(app)
```

Full Code for New Project Structure

`./netlify/function/heroes/model.js`

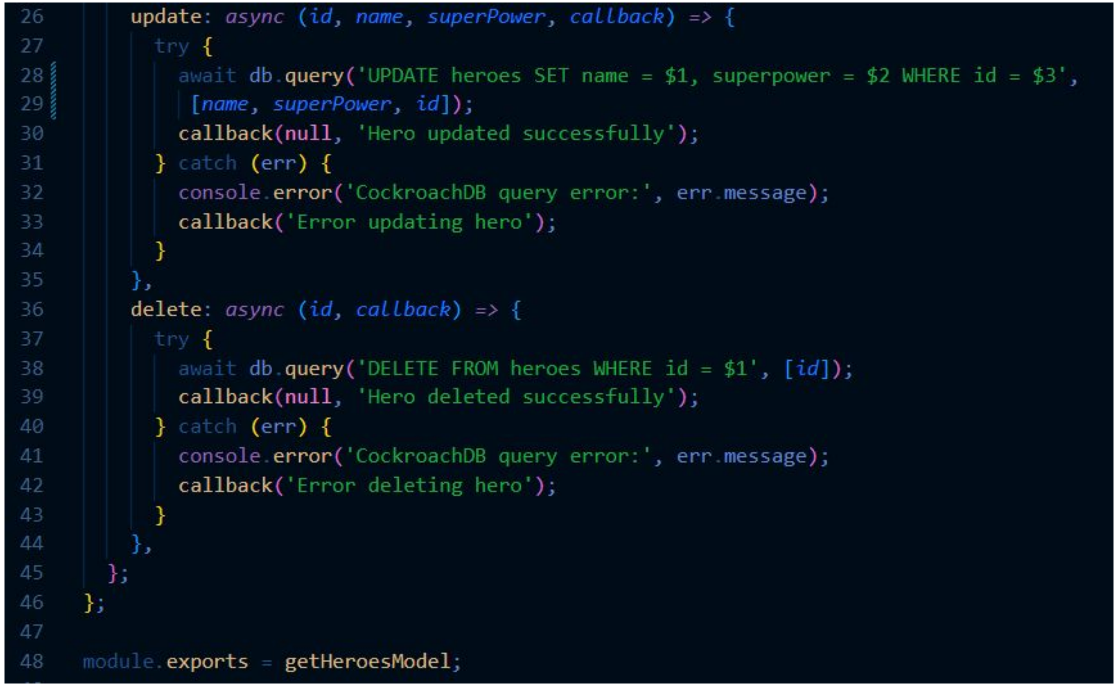


netlify > functions > heroes > model.js > ...

```javascript
const connectToDB = require('../../../src/db');

const getHeroesModel = () => {
  const db = connectToDB();

  return {
    findAll: async (callback) => {
      try {
        const result = await db.query('SELECT * FROM heroes');
        callback(result.rows);
      } catch (err) {
        console.error('CockroachDB query error:', err.message);
        callback([]);
      }
    },
    create: async (name, superPower, callback) => {
      try {
        await db.query('INSERT INTO heroes (name, superpower) VALUES ($1, $2)',
        [name, superPower]);
        callback(null, 'Hero created successfully');
      } catch (err) {
        console.error('CockroachDB query error:', err.message);
        callback('Error creating hero');
      }
    },
```

Full Code for New Project Structure

./netlify/function/heroes/model.js

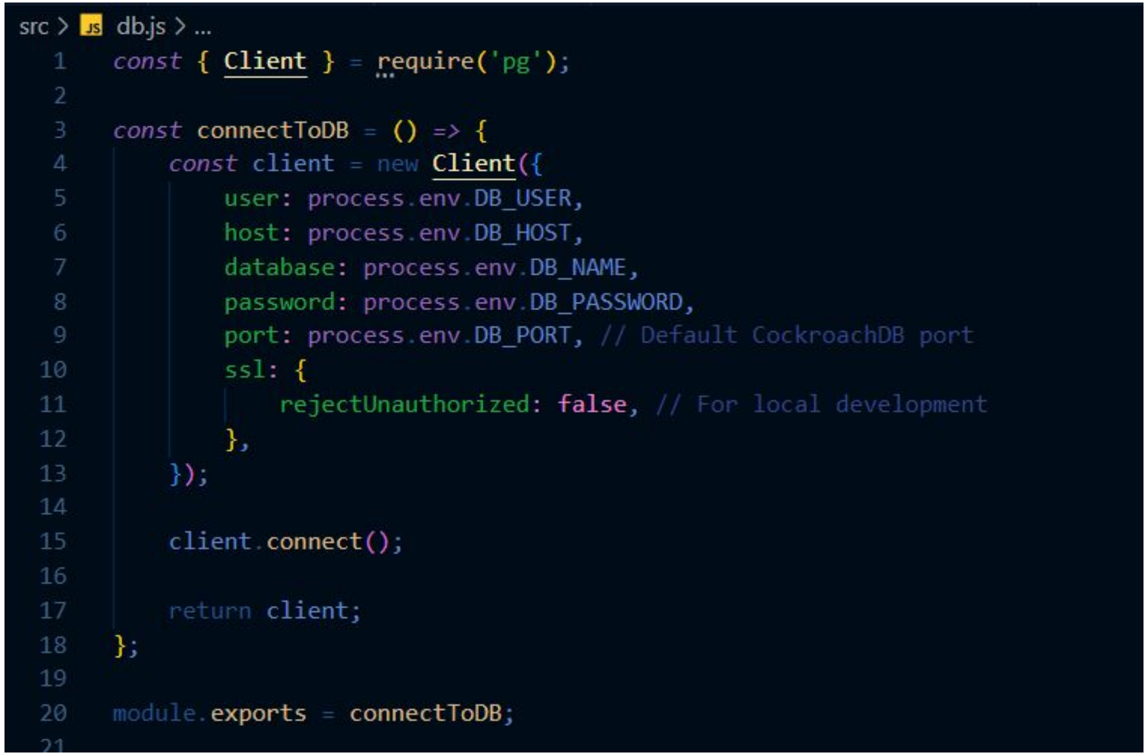


```javascript
    update: async (id, name, superPower, callback) => {
      try {
        await db.query('UPDATE heroes SET name = $1, superpower = $2 WHERE id = $3',
         [name, superPower, id]);
        callback(null, 'Hero updated successfully');
      } catch (err) {
        console.error('CockroachDB query error:', err.message);
        callback('Error updating hero');
      }
    },
    delete: async (id, callback) => {
      try {
        await db.query('DELETE FROM heroes WHERE id = $1', [id]);
        callback(null, 'Hero deleted successfully');
      } catch (err) {
        console.error('CockroachDB query error:', err.message);
        callback('Error deleting hero');
      }
    },
  };
};

module.exports = getHeroesModel;
```

Full Code for New Project Structure

`./src/db.js`

This file exports an async function connectToDB which connects to a CockroachDB. The connection URL and database name are read from the .env file.



src > db.js > ...

```javascript
const { Client } = require('pg');

const connectToDB = () => {
    const client = new Client({
        user: process.env.DB_USER,
        host: process.env.DB_HOST,
        database: process.env.DB_NAME,
        password: process.env.DB_PASSWORD,
        port: process.env.DB_PORT, // Default CockroachDB port
        ssl: {
            rejectUnauthorized: false, // For local development
        },
    });

    client.connect();

    return client;
};

module.exports = connectToDB;
```

Full Code for New Project Structure

`./.env`

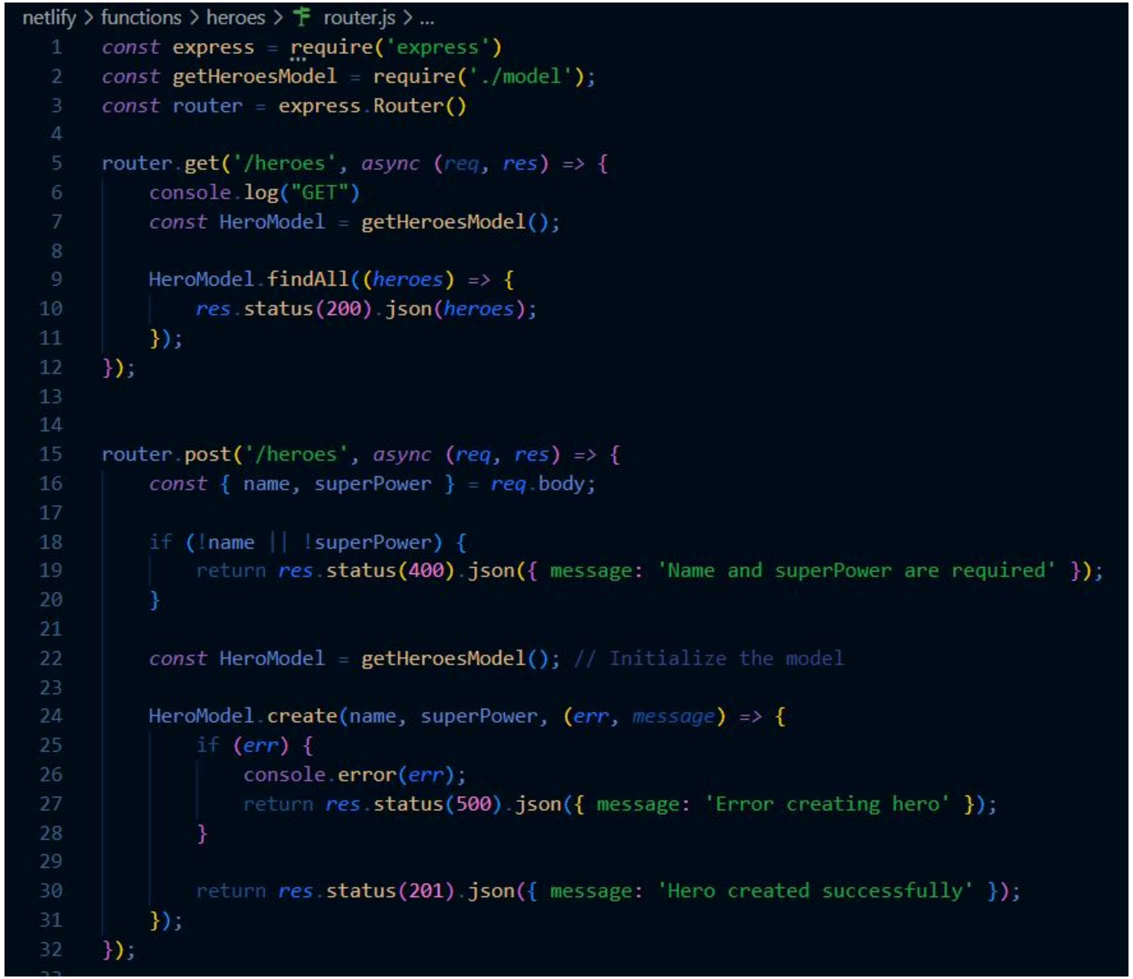
It is important to note that the '.env' file contains sensitive information, including the database connection URL and database name.

```
DB_USER=your_db_user
DB_HOST=your_db_host
DB_NAME=your_db_name
DB_PASSWORD=your_db_passsword
DB_PORT=26257
```

Full Code for New Project Structure

`./netlify/function/heroes/router.js`

This file defines the routing logic for the Netlify function. It requires the express and router modules, as well as the getHeroesModel function from model.js and the connectToDB function from 'src/db.js'.

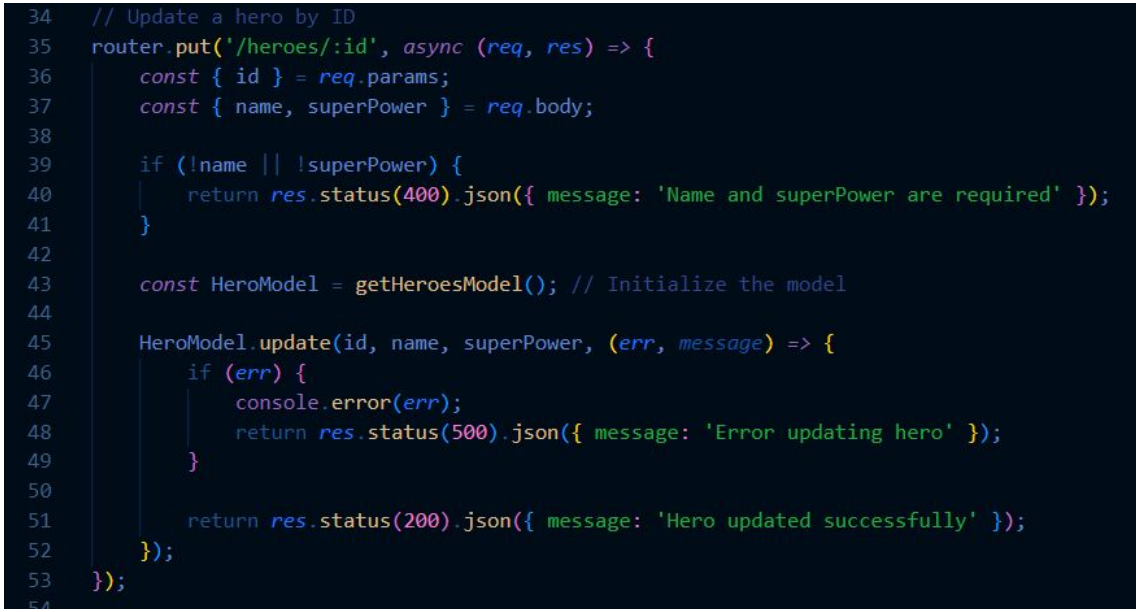


```javascript
const express = require('express')
const getHeroesModel = require('./model');
const router = express.Router()

router.get('/heroes', async (req, res) => {
    console.log("GET")
    const HeroModel = getHeroesModel();

    HeroModel.findAll((heroes) => {
        res.status(200).json(heroes);
    });
});


router.post('/heroes', async (req, res) => {
    const { name, superPower } = req.body;

    if (!name || !superPower) {
        return res.status(400).json({ message: 'Name and superPower are required' });
    }

    const HeroModel = getHeroesModel(); // Initialize the model

    HeroModel.create(name, superPower, (err, message) => {
        if (err) {
            console.error(err);
            return res.status(500).json({ message: 'Error creating hero' });
        }

        return res.status(201).json({ message: 'Hero created successfully' });
    });
});
```

Full Code for New Project Structure

`./netlify/function/heroes/router.js`

This file defines the routing logic for the Netlify function. It requires the express and router modules, as well as the getHeroesModel function from model.js and the connectToDB function from 'src/db.js'.

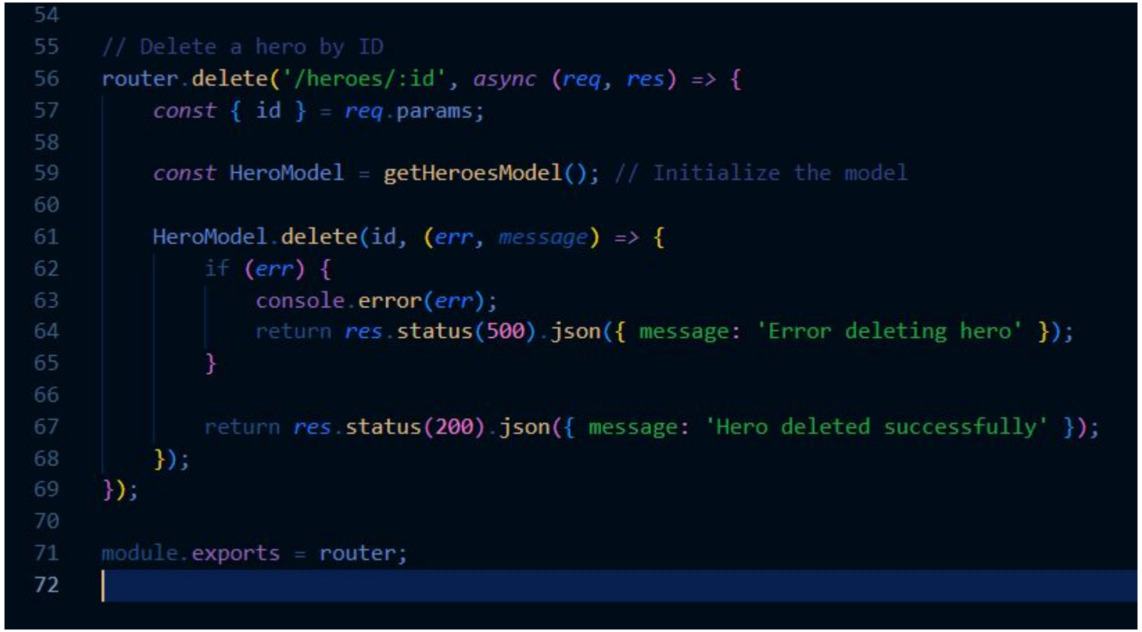


```javascript
// Update a hero by ID
router.put('/heroes/:id', async (req, res) => {
    const { id } = req.params;
    const { name, superPower } = req.body;

    if (!name || !superPower) {
        return res.status(400).json({ message: 'Name and superPower are required' });
    }

    const HeroModel = getHeroesModel(); // Initialize the model

    HeroModel.update(id, name, superPower, (err, message) => {
        if (err) {
            console.error(err);
            return res.status(500).json({ message: 'Error updating hero' });
        }

        return res.status(200).json({ message: 'Hero updated successfully' });
    });
});
```

Full Code for New Project Structure

`./netlify/function/heroes/router.js`

This file defines the routing logic for the Netlify function. It requires the express and router modules, as well as the getHeroesModel function from model.js and the connectToDB function from 'src/db.js'.



```javascript

// Delete a hero by ID
router.delete('/heroes/:id', async (req, res) => {
    const { id } = req.params;

    const HeroModel = getHeroesModel(); // Initialize the model

    HeroModel.delete(id, (err, message) => {
        if (err) {
            console.error(err);
            return res.status(500).json({ message: 'Error deleting hero' });
        }

        return res.status(200).json({ message: 'Hero deleted successfully' });
    });
});

module.exports = router;

```

Netlify CLI

Netlify CLI

To run this project locally, you'll need to have Node.js and the Netlify CLI installed. You can install Node.js from the official website, and you can install the Netlify CLI using NPM:

```
npm install netlify-cli -g
```

```
npm install netlify-cli -g --unsafe-perm=true --allow-root
```

By default, Netlify collects data on usage of Netlify CLI commands, opt out of sharing usage data with the command line:

```
netlify --telemetry-disable
```

Netlify CLI

⚠️ Some users reporting an error related to Execution Policies. This is because the script execution policies in PowerShell may be set to a restricted mode by default, preventing the running of scripts.

```
File C:\Users\your-user-name\AppData\Roaming\npm\netlify.ps1 cannot be loaded because running scripts is disabled
on this system. For more information, see about_Execution_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.
```

To solve the "Execution Policies" error, you need to run the following command in the Terminal:

```
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy Unrestrict
```

Run the project

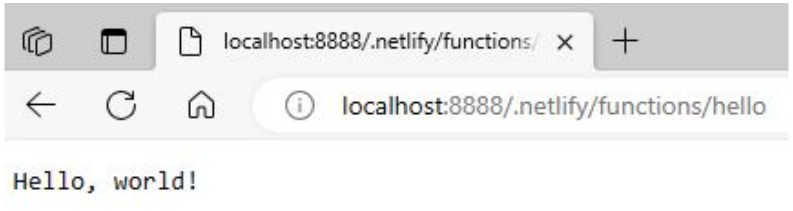
Once you have the dependencies installed, you can run the project locally using the following command:

```
netlify dev
```

Serverless Functions API Urls

When you deploy a serverless function to Netlify, it becomes available via a unique URL that includes the prefix "/.netlify/functions/".

So if you have a function called "hello" in your Netlify project, you can access it via the URL "/.netlify/functions/hello".

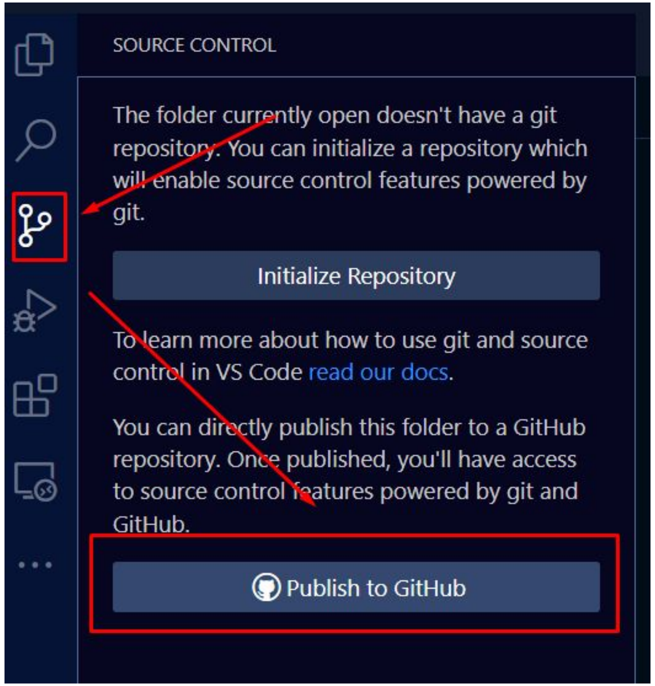

localhost:8888/.netlify/functions/
localhost:8888/.netlify/functions/hello

```
Hello, world!
```

GitHub Repository

Create a Private GitHub Repository for Source Control and publish branch

1



SOURCE CONTROL

The folder currently open doesn't have a git repository. You can initialize a repository which will enable source control features powered by git.

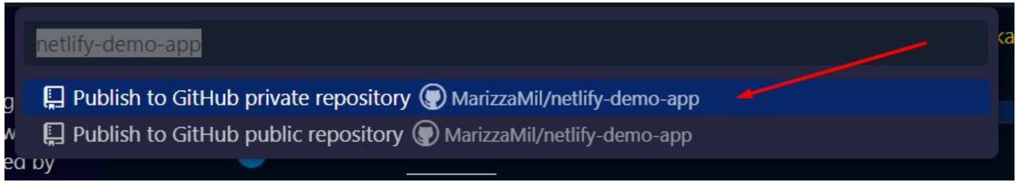
Initialize Repository

To learn more about how to use git and source control in VS Code read our docs.

You can directly publish this folder to a GitHub repository. Once published, you'll have access to source control features powered by git and GitHub.

Publish to GitHub

2

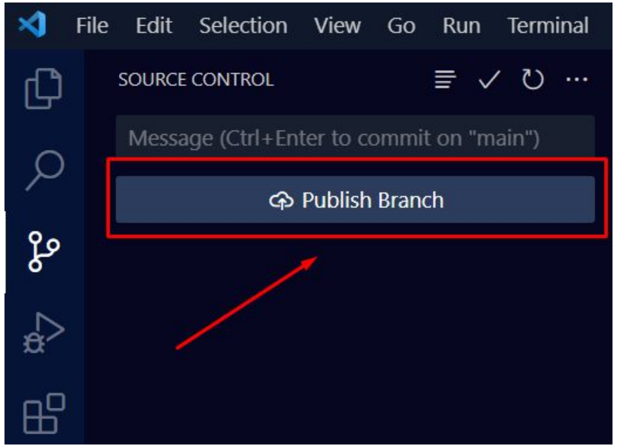


netlify-demo-app

Publish to GitHub private repository MarizzaMil/netlify-demo-app

Publish to GitHub public repository MarizzaMil/netlify-demo-app

3



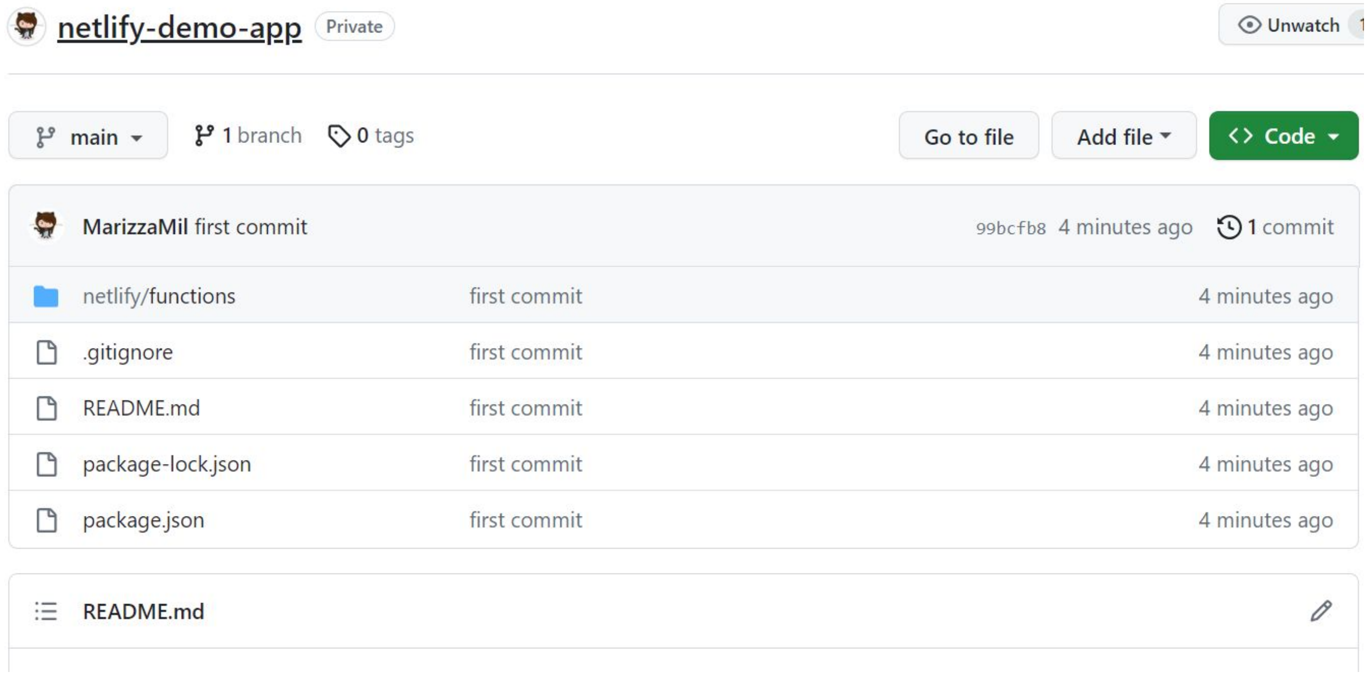
File Edit Selection View Go Run Terminal

SOURCE CONTROL

Message (Ctrl+Enter to commit on "main")

Publish Branch

Create a Private GitHub Repository for Source Control



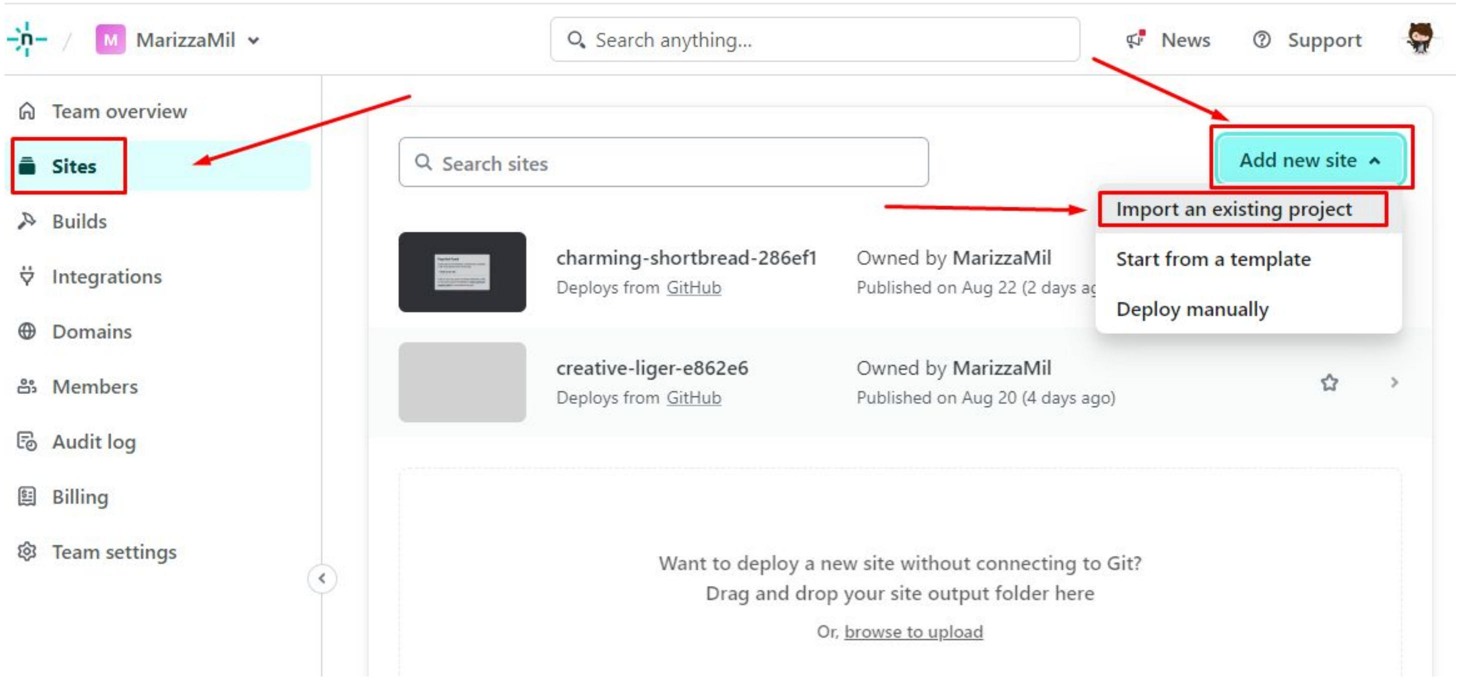
MarizzaMil first commit 99bcfb8 4 minutes ago 1 commit

netlify/functions	first commit	4 minutes ago
.gitignore	first commit	4 minutes ago
README.md	first commit	4 minutes ago
package-lock.json	first commit	4 minutes ago
package.json	first commit	4 minutes ago

README.md

Deploy Project in the Netlify

Connecting Netlify to Your GitHub Account



charming-shortbread-286ef1
Deploys from GitHub
Owned by MarizzaMil
Published on Aug 22 (2 days ag

creative-liger-e862e6
Deploys from GitHub
Owned by MarizzaMil
Published on Aug 20 (4 days ago)

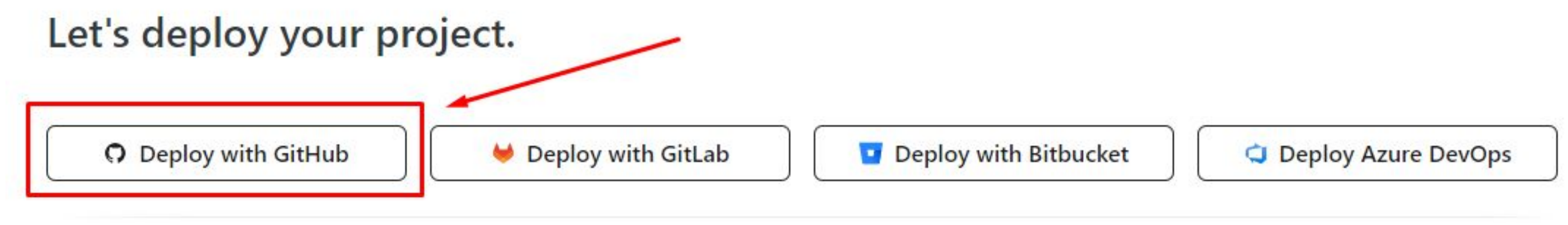
Want to deploy a new site without connecting to Git?
Drag and drop your site output folder here
Or, browse to upload

Connecting Netlify to Your GitHub Account

3. Select "GitHub" as your Git provider.
4. Follow the prompts to connect your GitHub account to Netlify. You may need to grant Netlify access to your GitHub account.

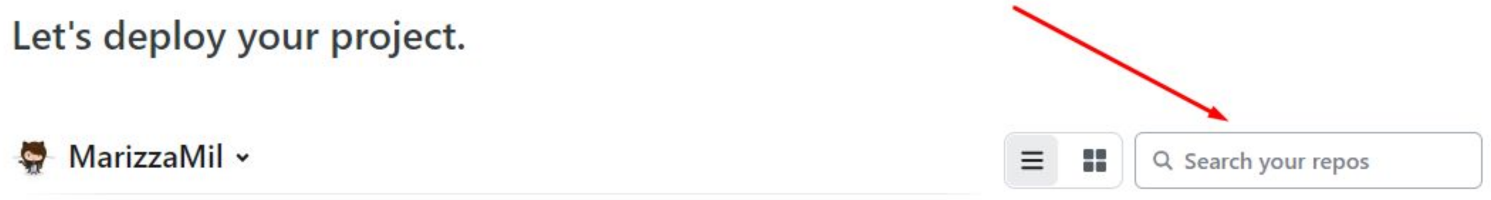
Choosing "Only select repositories" option when connecting Netlify to GitHub means that you are selecting specific repositories from your GitHub account that you want to connect with Netlify. This provides more control and security to your deployment process.

Connect to Git provider

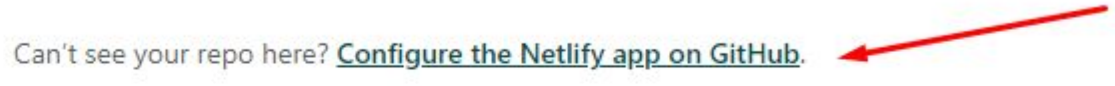

Let's deploy your project.

Select repository

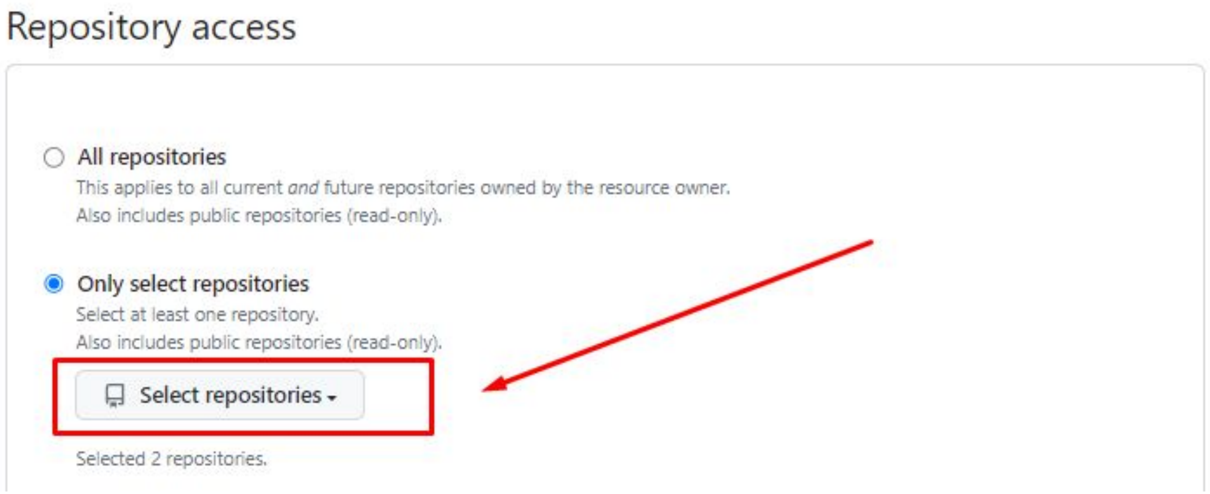
Once your GitHub account is connected, you can pick a repository to deploy to Netlify. In the "Pick a repository" section, select the repository you want to deploy.


Let's deploy your project.

If you have created a GitHub repository for your Netlify project, but it's not appearing in the list of available repositories on Netlify, you may need to configure the Netlify app on GitHub. Here's how to do it:


Can't see your repo here? Configure the Netlify app on GitHub.

Configure site and deploy



Repository access

[ ] All repositories

This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

[✓ Only select repositories]

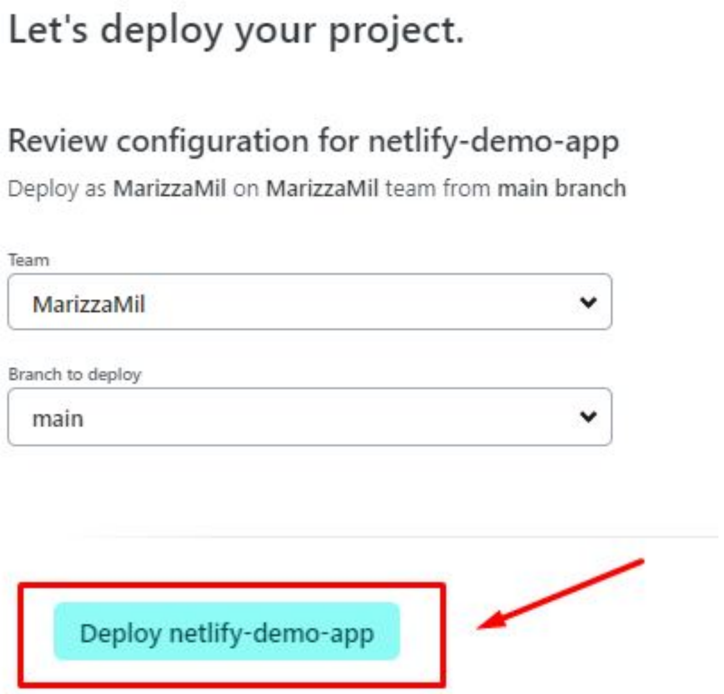
Select at least one repository.
Also includes public repositories (read-only).

Selected 2 repositories.

In the "Repository access" section, if you chose "Only select repositories" choose specific repositories. Select the repository you want to link to your Netlify site. Click "Save" to apply the changes.

Configure site and deploy

5. Configure your build settings. This will depend on the type of project you're deploying.
6. Click "Deploy site" to start the deployment process.


Let's deploy your project.

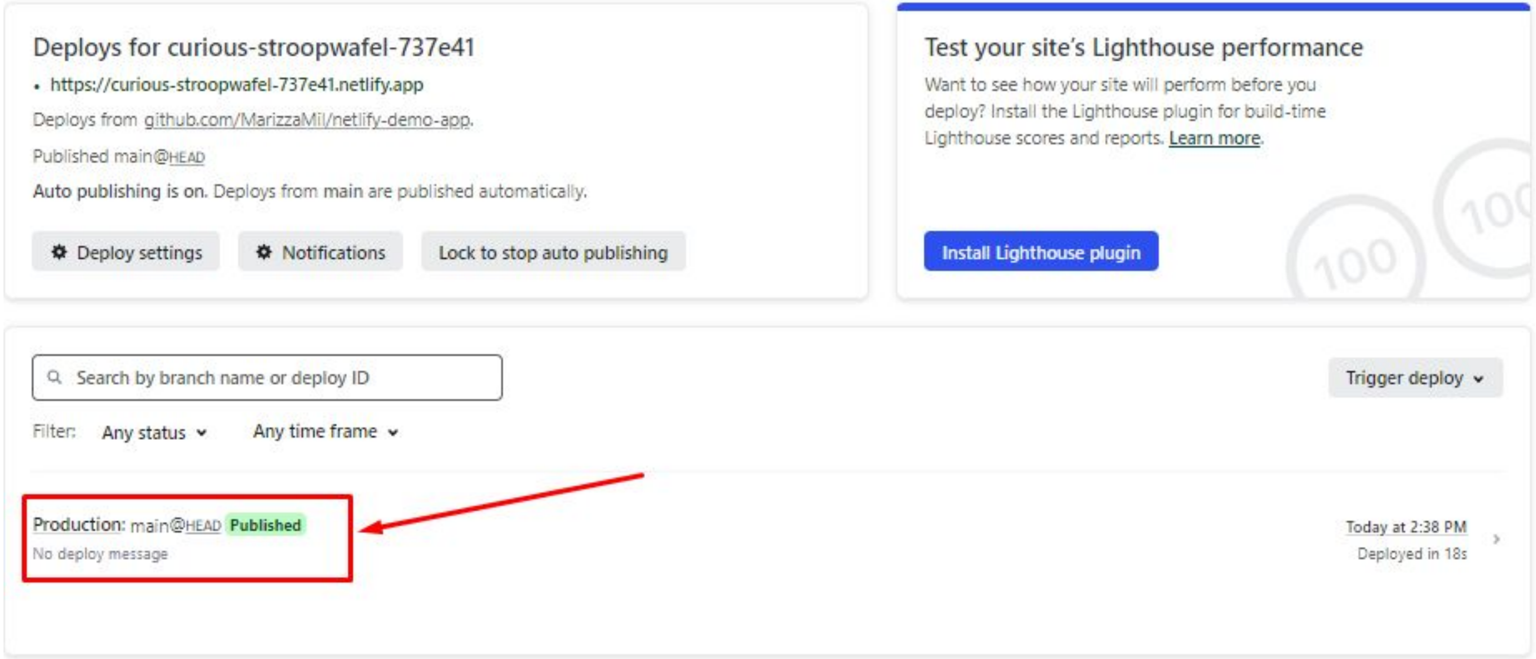
Review configuration for netlify-demo-app

Deploy as **MarizzaMil** on **MarizzaMil** team from **main branch**

Team: MarizzaMil

Branch to deploy: main

Production



Deploys for curious-stroopwafel-737e41

- https://curious-stroopwafel-737e41.netlify.app

Deploys from github.com/MarizzaMil/netlify-demo-app.

Published main@HEAD

Auto publishing is on. Deploys from main are published automatically.

Test your site's Lighthouse performance

Want to see how your site will perform before you deploy? Install the Lighthouse plugin for build-time Lighthouse scores and reports. Learn more.

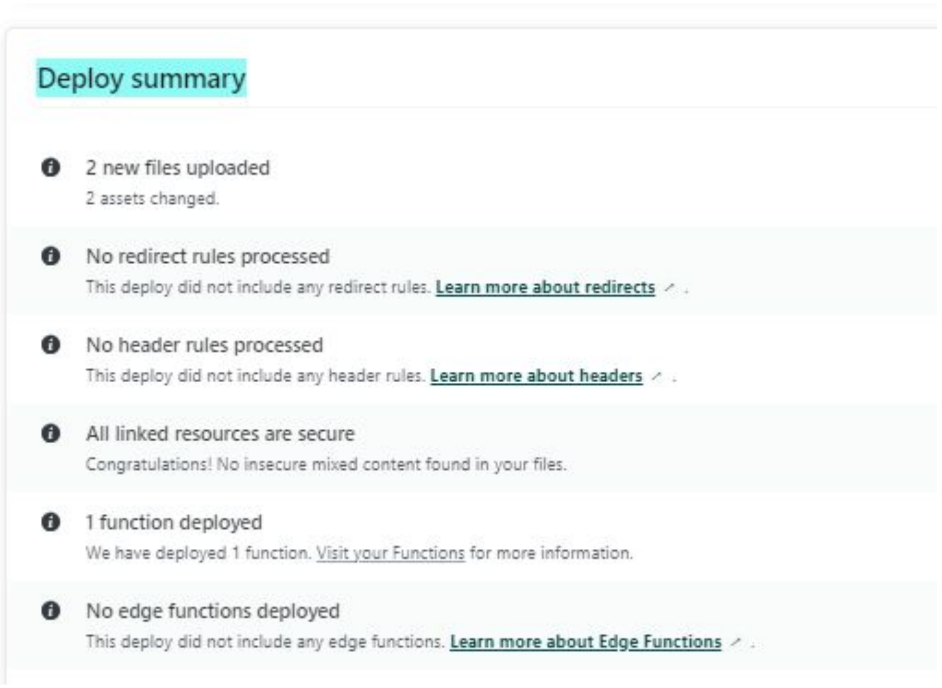
Production: main@HEAD Published

No deploy message

Today at 2:38 PM

Deployed in 18s

Deploy summary



Deploy summary

2 new files uploaded
2 assets changed.

No redirect rules processed
This deploy did not include any redirect rules. Learn more about redirects .

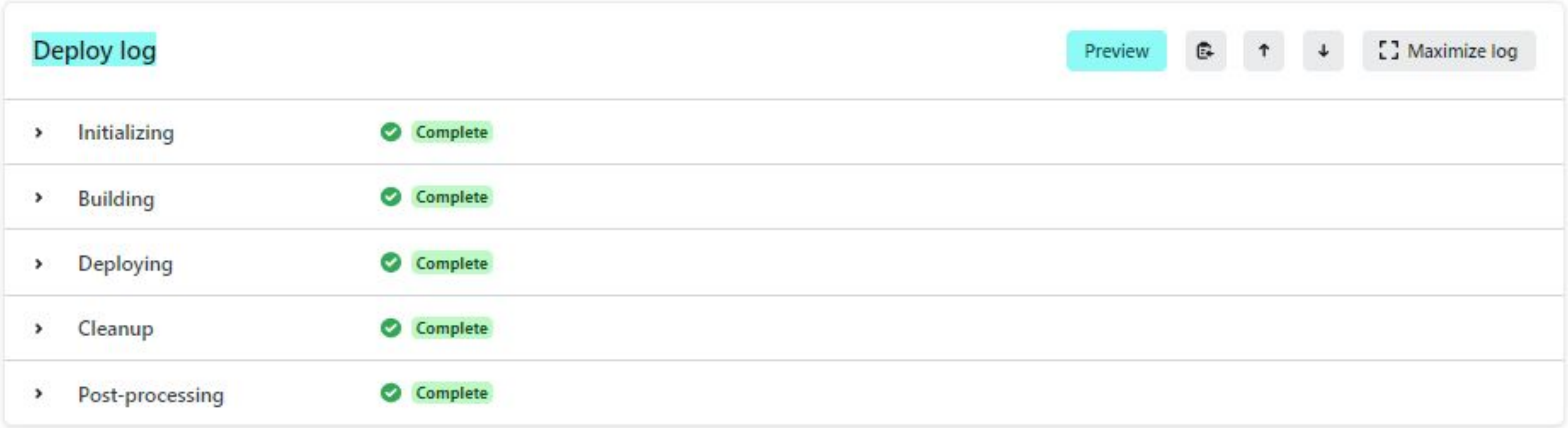
No header rules processed
This deploy did not include any header rules. Learn more about headers .

All linked resources are secure
Congratulations! No insecure mixed content found in your files.

1 function deployed
We have deployed 1 function. Visit your Functions for more information.

No edge functions deployed
This deploy did not include any edge functions. Learn more about Edge Functions .

Deploy log

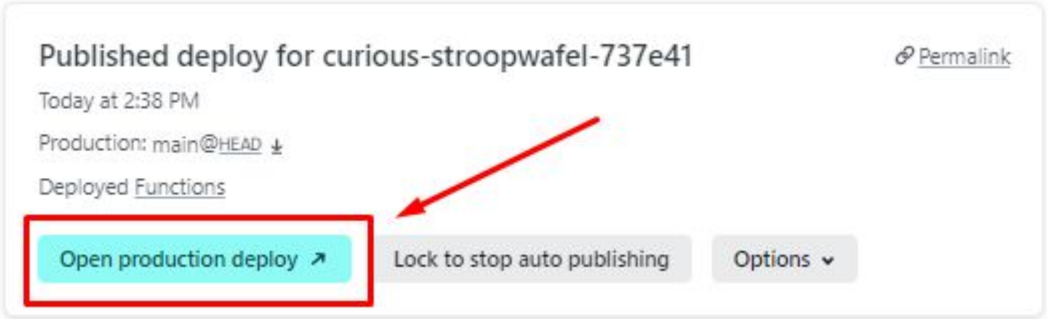


Deploy log

Preview Maximize log

Initializing	✓ Complete
Building	✓ Complete
Deploying	✓ Complete
Cleanup	✓ Complete
Post-processing	✓ Complete

Production



Published deploy for curious-stroopwafel-737e41

Permalink

Today at 2:38 PM

Production: main@HEAD

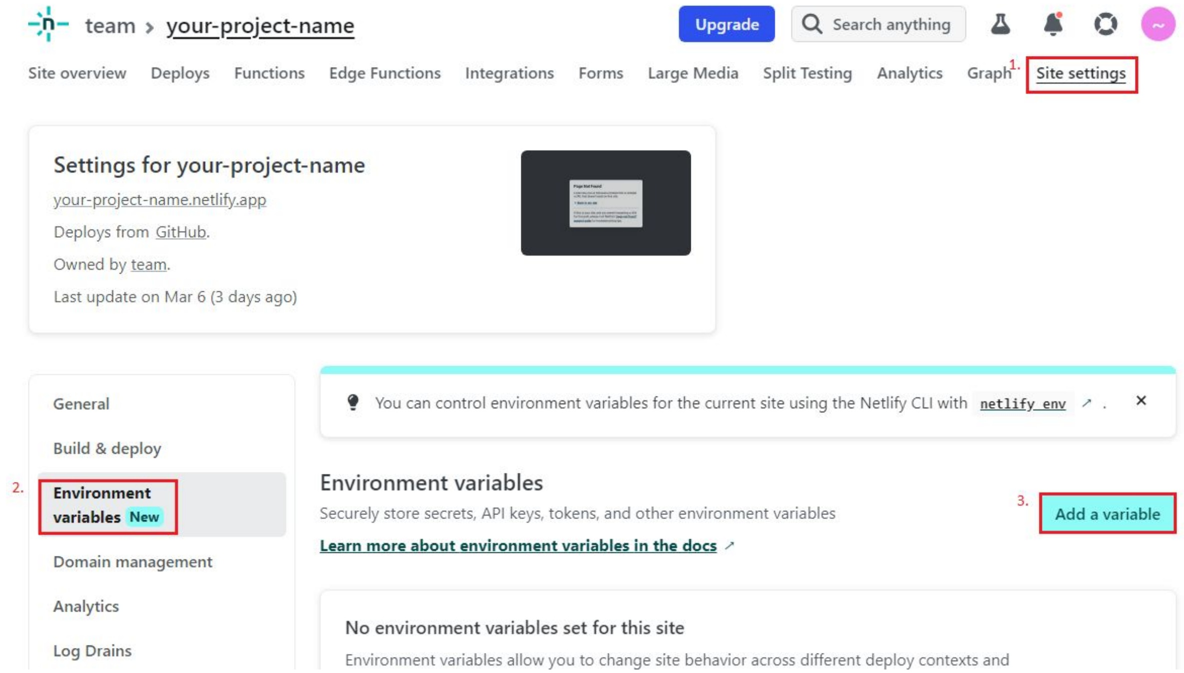
Deployed Functions

Environment Variables on Netlify

The .env file is used to define environment variables locally, which are loaded into the application at runtime. However, when deploying to a serverless platform like Netlify, the .env file is not loaded automatically.

To configure environment variables on Netlify, you can use the Netlify Environment Variables feature. This allows you to define environment variables in the Netlify dashboard, which can then be used in your serverless functions.

Environment Variables on Netlify



Settings for your-project-name

your-project-name.netlify.app

Deploys from GitHub.

Owned by team.

Last update on Mar 6 (3 days ago)

You can control environment variables for the current site using the Netlify CLI with `netlify env`.

Environment variables

Securely store secrets, API keys, tokens, and other environment variables

Learn more about environment variables in the docs

No environment variables set for this site

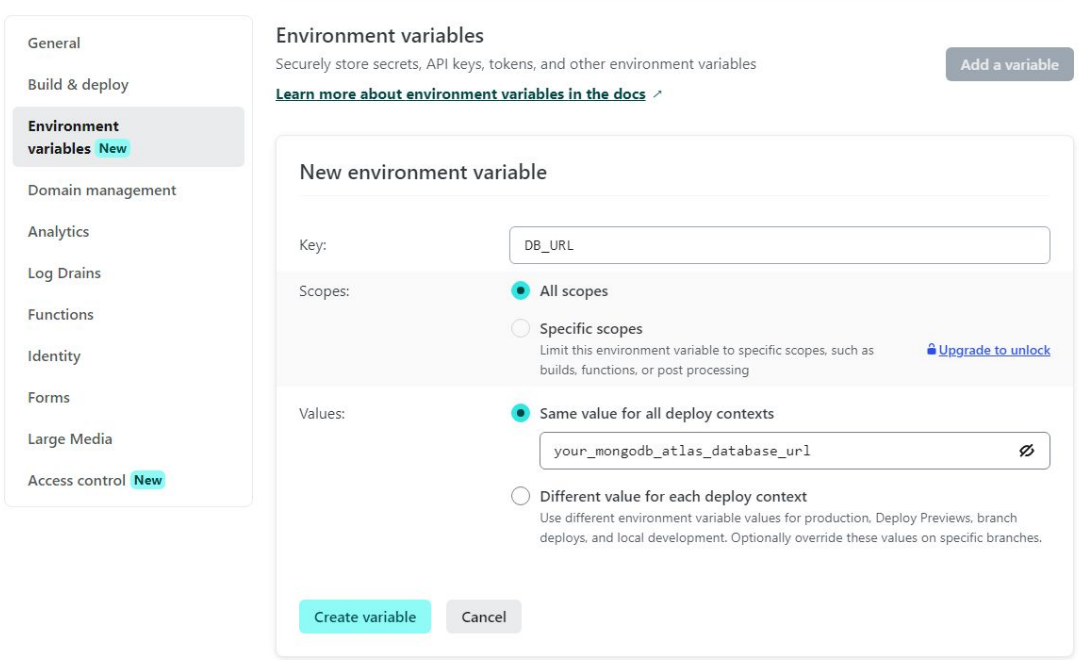
Environment variables allow you to change site behavior across different deploy contexts and

Environment Variables on Netlify

To do this:

1. Go to your Netlify site dashboard.
2. Click on "Site settings".
3. On the "General" menu on the left-hand side, click on "Environment variables".
4. Click "Add a variables" and add the necessary environment variables.
5. Save your changes.

Environment Variables on Netlify



Environment variables

Securely store secrets, API keys, tokens, and other environment variables

Learn more about environment variables in the docs ↗

New environment variable

Key: DB_URL

Scopes: [✓ All scopes] Specific scopes

Limit this environment variable to specific scopes, such as builds, functions, or post processing

Values: [✓ Same value for all deploy contexts]

```
your_mongodb_atlas_database_url
```

Different value for each deploy context

Use different environment variable values for production, Deploy Previews, branch deploys, and local development. Optionally override these values on specific branches.

Summary

This tutorial covered how to build a serverless API using Netlify Functions and CockroachDB. We went through the process of setting up the CockroachDB connecting to the database from our Netlify Functions, and creating a simple API.

We started by setting up the CockroachDB account and created a new database. We then created a Netlify site and added environment variables for the database connection details. We installed the necessary dependencies for our project, including Express, and Serverless HTTP.

Next, we created a 'model.js' file to define our data schema, and a 'router.js' file to define the routes for our API. We then created an 'index.js' file that imports the router and sets up middleware to restrict API access to specific hosts. We also created a 'db.js' file to connect to our CockroachDB database.

After creating our files, we tested our API by sending requests to the endpoints. We used the Chrome browser to test the API endpoints and verified that the actions were being incremented in the CockroachDB database.

I recommend trying to write the code yourself as it can help you better understand the concepts and techniques used in this tutorial. Don't worry if you make mistakes or encounter errors, it's all part of the learning process. Feel free to experiment and modify the code to fit your needs. Good luck!