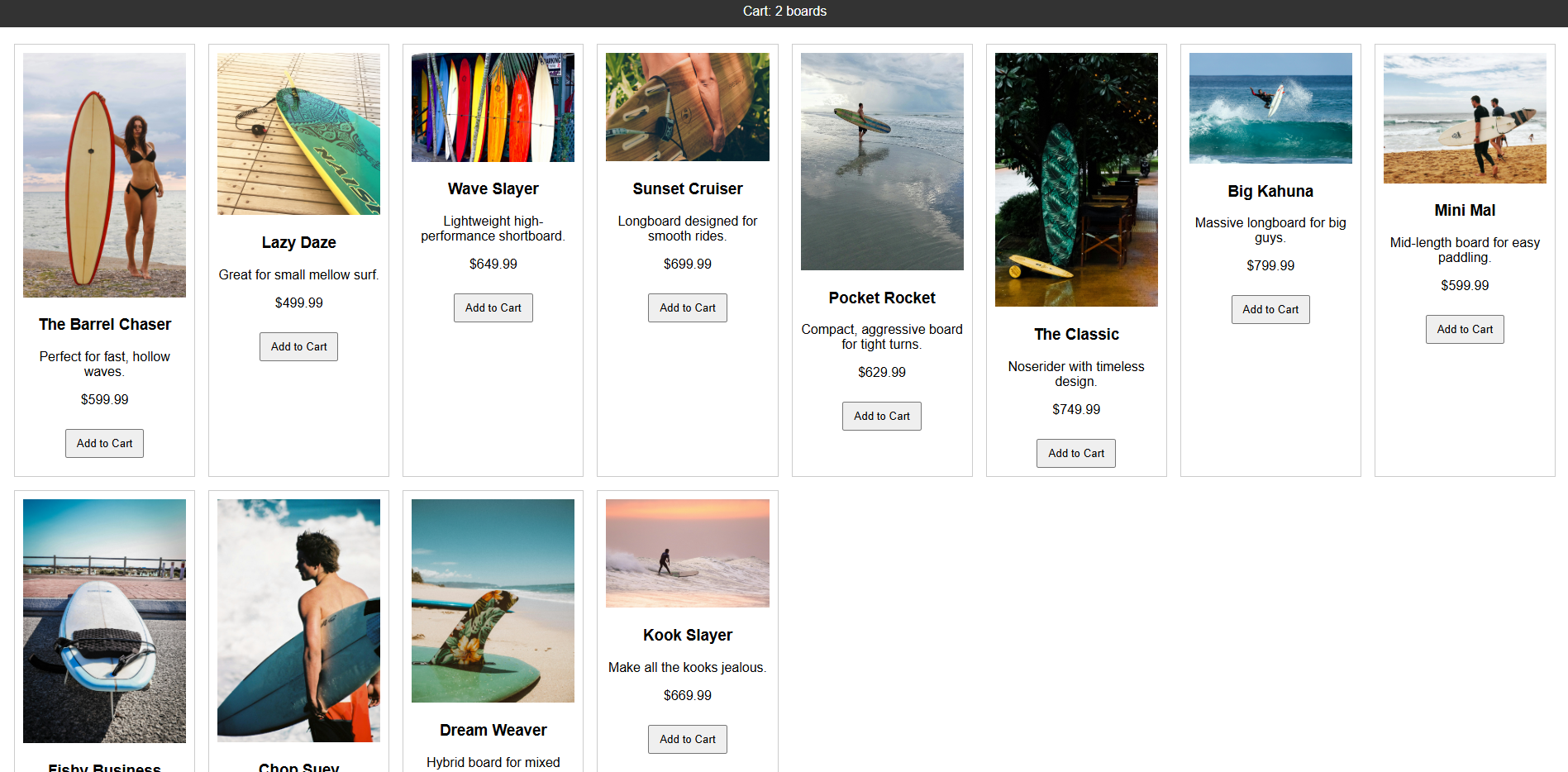
Cocky Kooks - Tutorial

A simple surfboard catalog app with a cart and a checkout form.

Built using vanilla JS, Node.js backend, and Postgres database.



# Table of Contents

[Table of Contents](#_bszcdeec70dh)

[Introduction](#_vx4affp8g64o)

[Checklist](#_fj4n3wl3hzry)

[Getting the Code](#_1lyh9hs2jxk8)

[Overview of Moving Parts](#_4gc09osobpw9)

[Database Setup](#_3ls3bv1o5aum)

[Install Postgres](#_dwwumtmbawej)

[Verify Installation with pgAdmin](#_h39qk5ioc5f5)

[Connect to the Database Server from pdAdmin](#_jkjasbcgeoiq)

[Create the Database](#_8yabnwdfqo3x)

[Create the Tables](#_y3xwonazr34k)

[Seed the Database](#_cgkq3of4zu8k)

[Backend Setup and Development](#_2lila9j4g30h)

[Install Node:](#_5foltxh40zuw)

[Setup the backend project](#_4e5n7mcbhhnd)

[Starter Code for the Backend Server](#_qtl20uxgj2bj)

[API Code for the Backend Server](#_yfy2czao9f97)

[Frontend Development](#_3i2iwv3yv6yf)

[Main surfboard catalog page](#_q70yqww0v6nz)

[CSS code](#_1yyg6ba16aaw)

[JavaScript Code](#_ciqxydie4isu)

[Cart Code](#_7mr415thwoip)

[HTML Code](#_mip65n6em2zd)

[Update your CSS code a bit](#_x0ipmc671hz)

[JavaScript Code](#_jj0lczupuh42)

[Running the Projects](#_4ycxp63ukmdv)

[Backend](#_8bug9a1ps1o9)

[Frontend](#_6568h5lusykr)

[Confirming Submissions](#_tbabqjj1mbh6)

# Introduction

In this project, you’ll create a simple surfboard catalog website where users can browse a list of boards, add them to their cart, and submit a purchase request. You'll build everything using **vanilla JavaScript**, a lightweight **Node.js backend**, and a **PostgreSQL** database to store and retrieve data.

This project is intentionally kept minimal and framework-free to help you focus on how the core web technologies connect. By the end, you’ll have hands-on experience with:

* Structuring an HTML/CSS/JavaScript project
* Making API calls using fetch()
* Managing state (like a shopping cart) in the browser
* Writing backend endpoints in Express (Node.js)
* Connecting to a Postgres database and running SQL
* Sending and storing form data via an API

# Checklist

Here’s some preliminary things you’ll need.

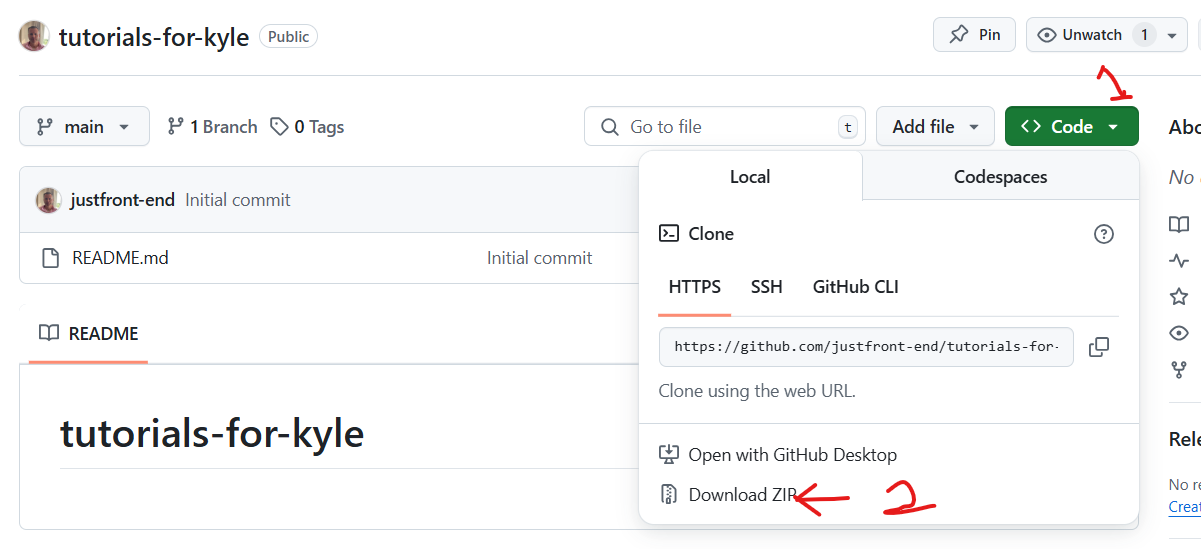
* VS Code: <https://code.visualstudio.com/>
* Node.js installed  
   [Download Node.js](https://nodejs.org/)
* PostgreSQL installed  
   [Download PostgreSQL](https://www.postgresql.org/download/)
* http-server installed

<https://www.npmjs.com/package/http-server>

# Getting the Code

<https://github.com/justfront-end/tutorials-for-kyle.git>

Click the green button, then click the Download ZIP text.



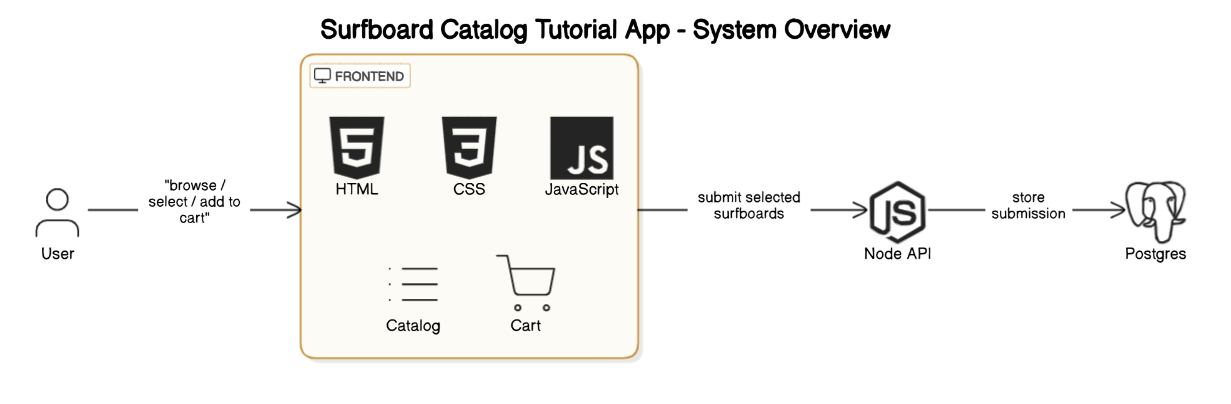
I can teach later how to use git. For now, just download the zip file and unzip it somewhere, anywhere, on your computer.

Here’s how you’d do it via the command line:

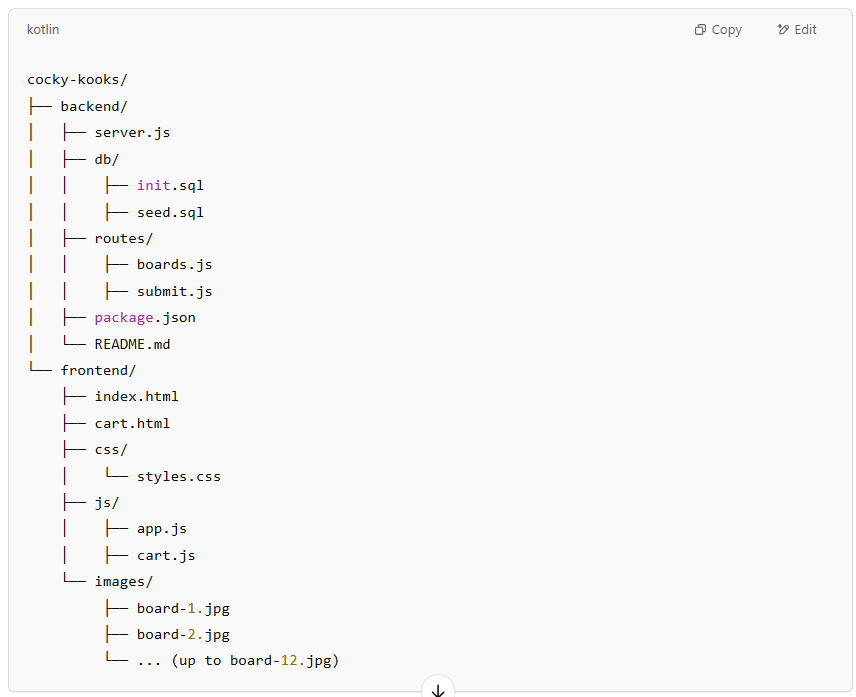
git clone <https://github.com/justfront-end/tutorials-for-kyle.git>

# Overview of Moving Parts

Big picture:



Files:



# Database Setup

## Install Postgres

<https://www.postgresql.org/download/>

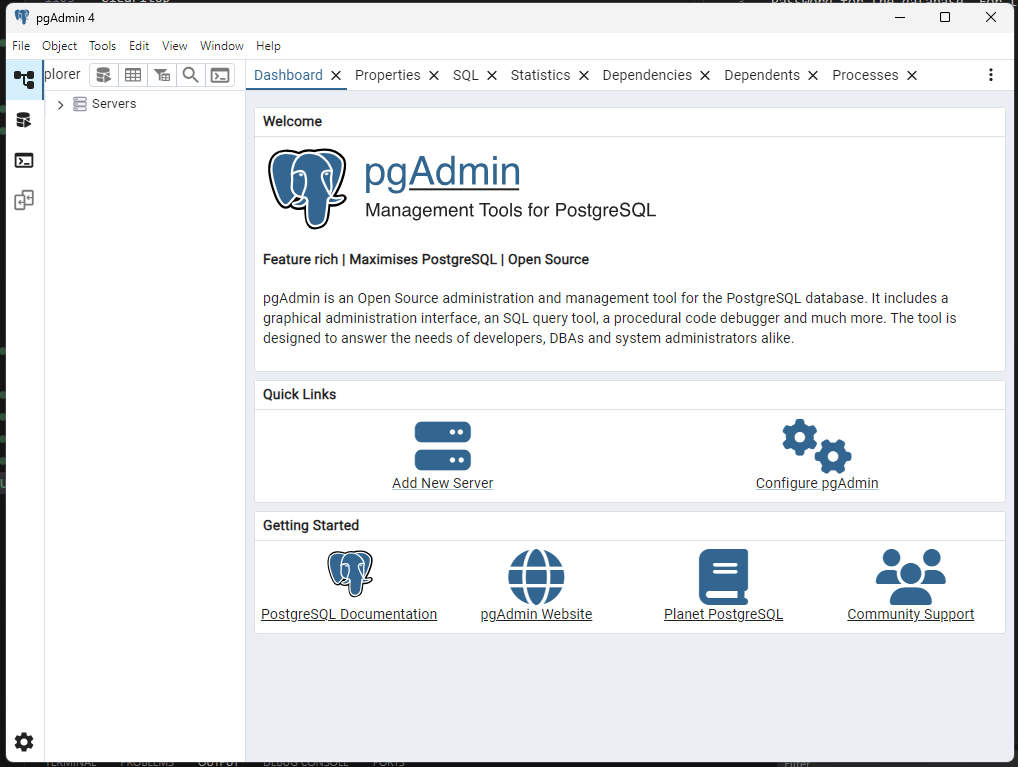
Pick the Mac or Windows installer. Download the latest version.

Install it.

## Verify Installation with pgAdmin

You’ll now have a new program named pgAdmin.

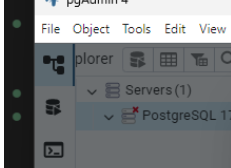
Looks like this:



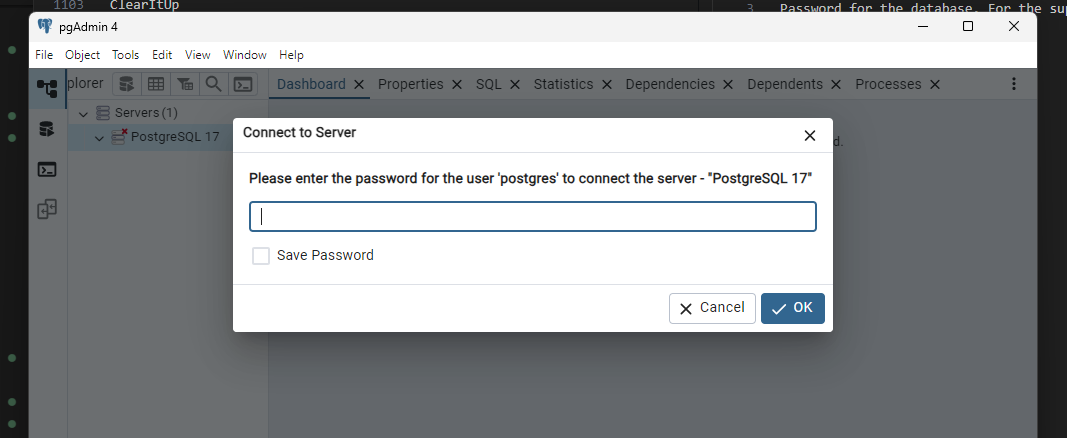
Confirm that you have pgAdmin.

## Connect to the Database Server from pdAdmin

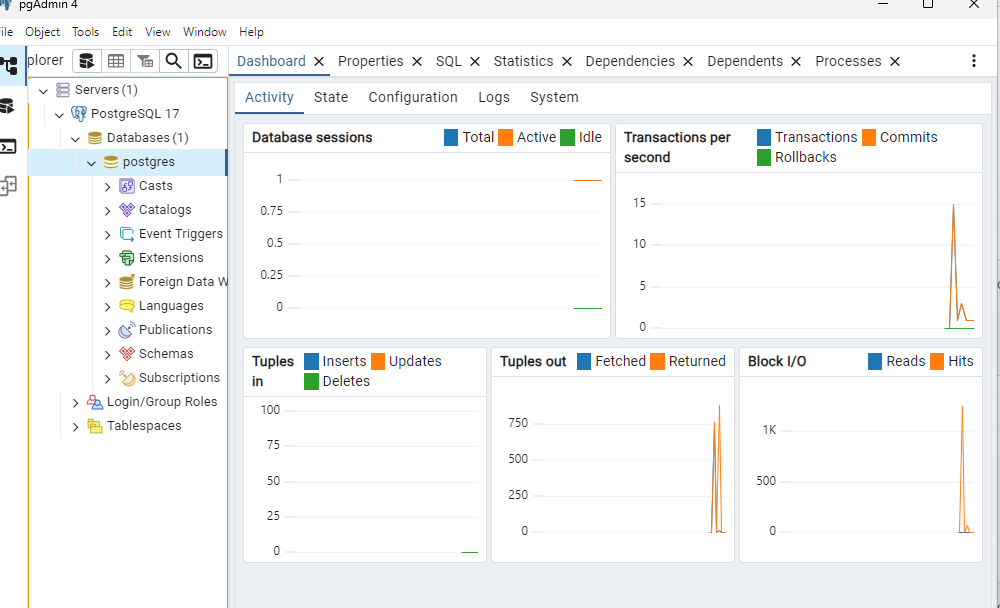
Toggle open this server button



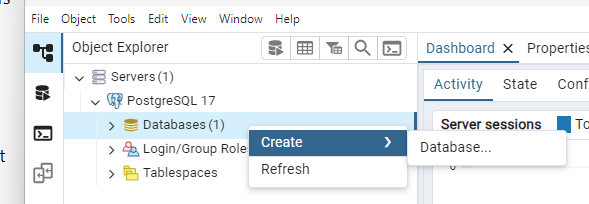
Then enter your superuser password, in this prompt. (pass is: admin)



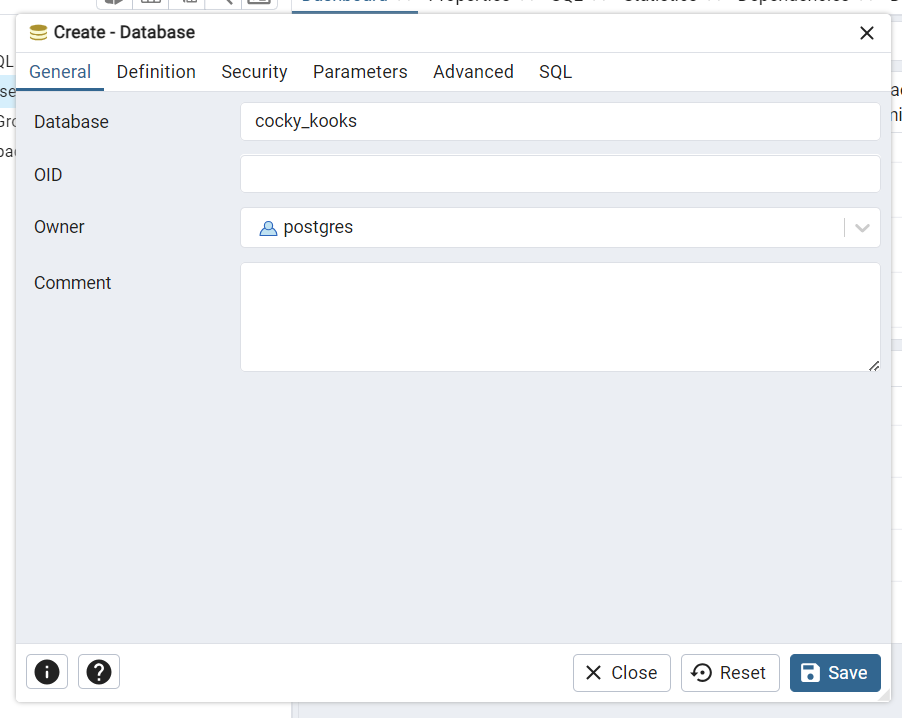
Now you’re connected to the database server.



## Create the Database



Right click the Databases folder, and do what the pic shows above.



We'll name the database cocky\_kooks. As shown above.

Click Save.

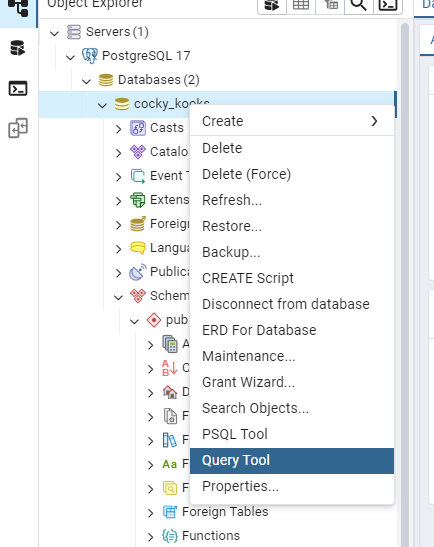
## Create the Tables

Note: You’ll find the SQL code for setting up the tables in this location:

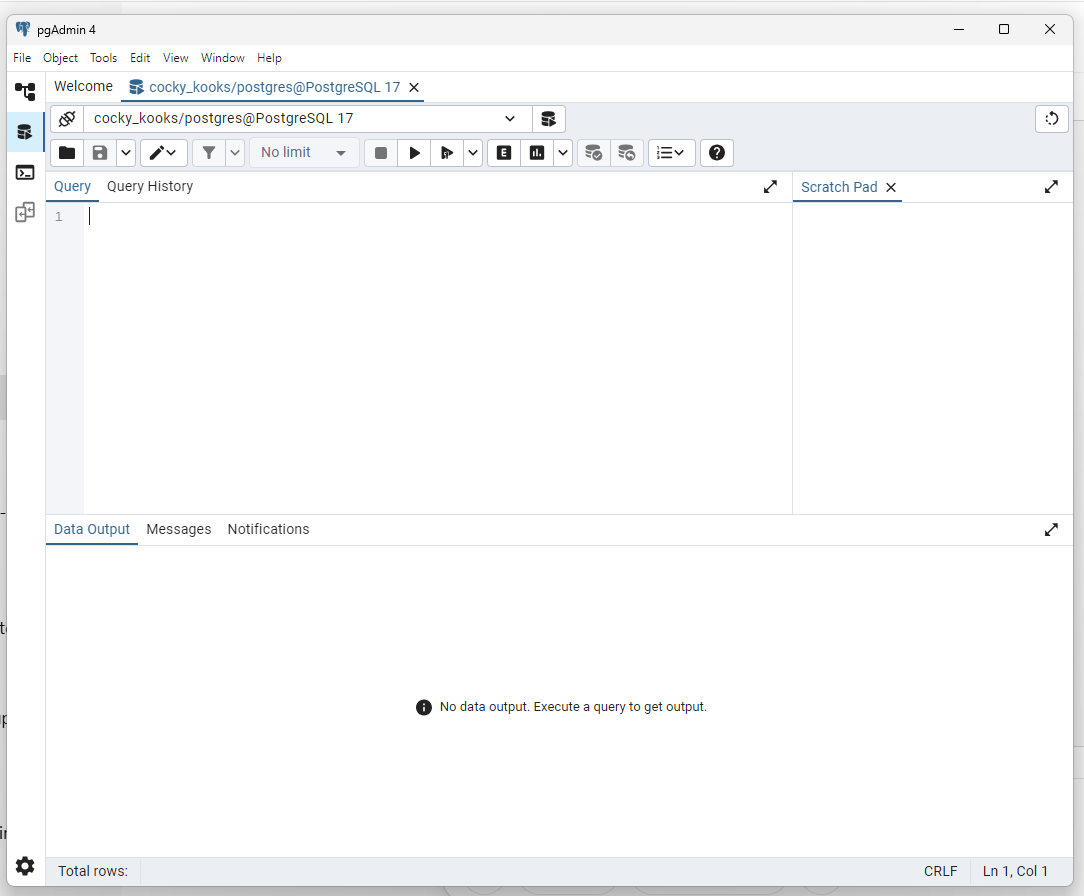
backend/db/init.sql

You can copy code from that file and paste it into the Query Tool.

Right click the cock\_kooks database, and click Query Tool.



A new window will appear, as shown:



Copy and paste the following code into the new window:

-- Create surfboards table

CREATE TABLE surfboards (

id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

price NUMERIC(10, 2) NOT NULL,

description TEXT,

image\_url TEXT

);

-- Create cart submissions table

CREATE TABLE cart\_submissions (

id SERIAL PRIMARY KEY,

name TEXT NOT NULL,

email TEXT NOT NULL,

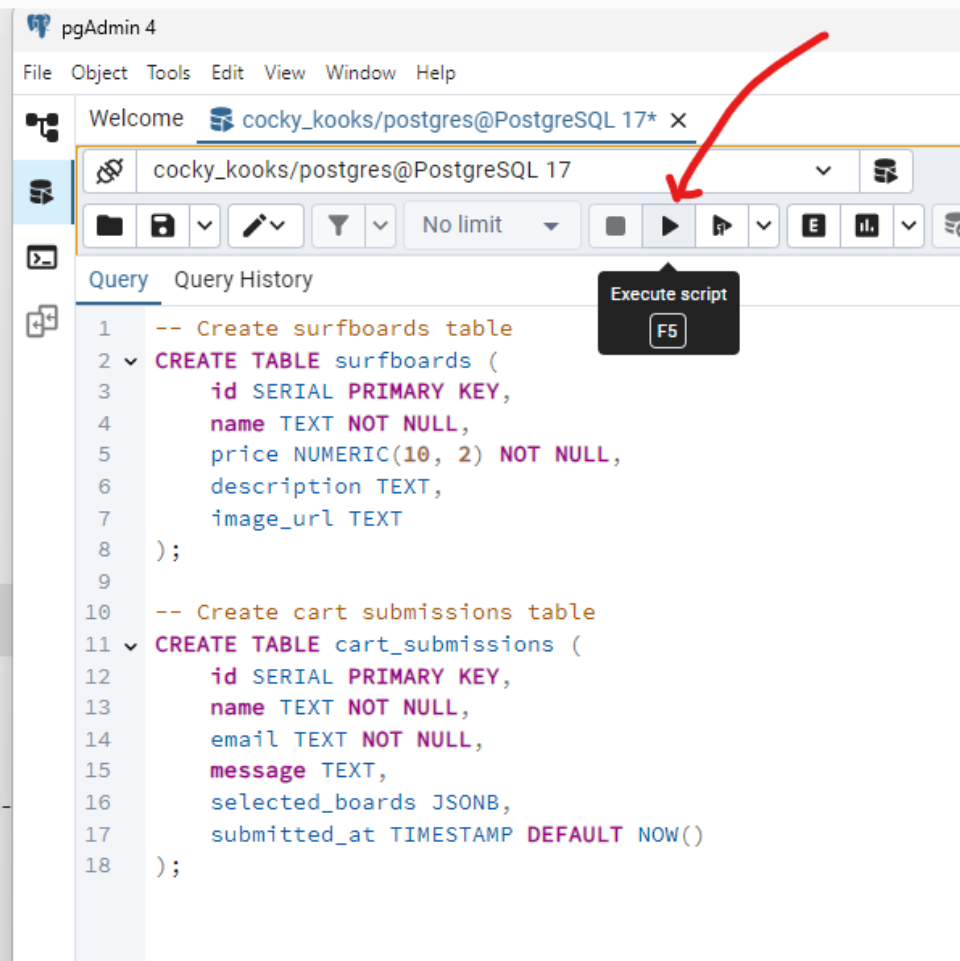
message TEXT,

selected\_boards JSONB,

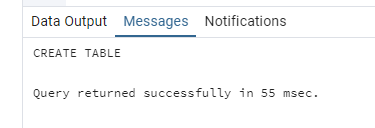
submitted\_at TIMESTAMP DEFAULT NOW()

);

Then click this button

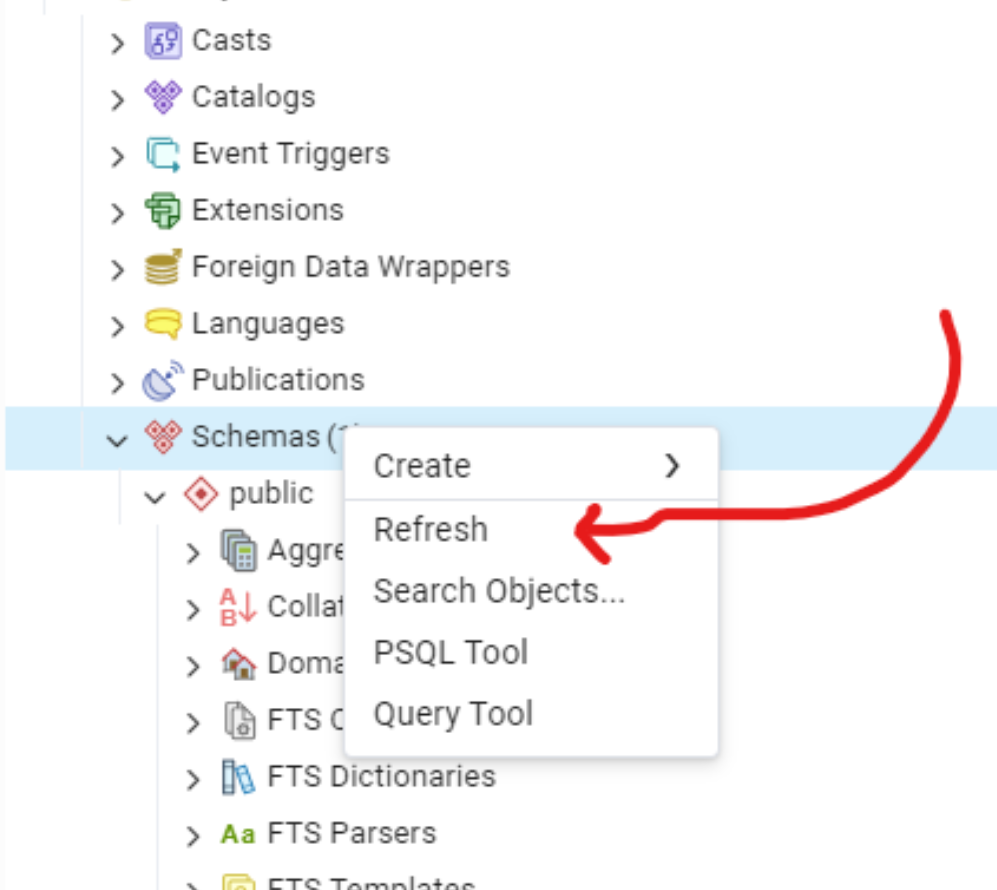


You should see this message:

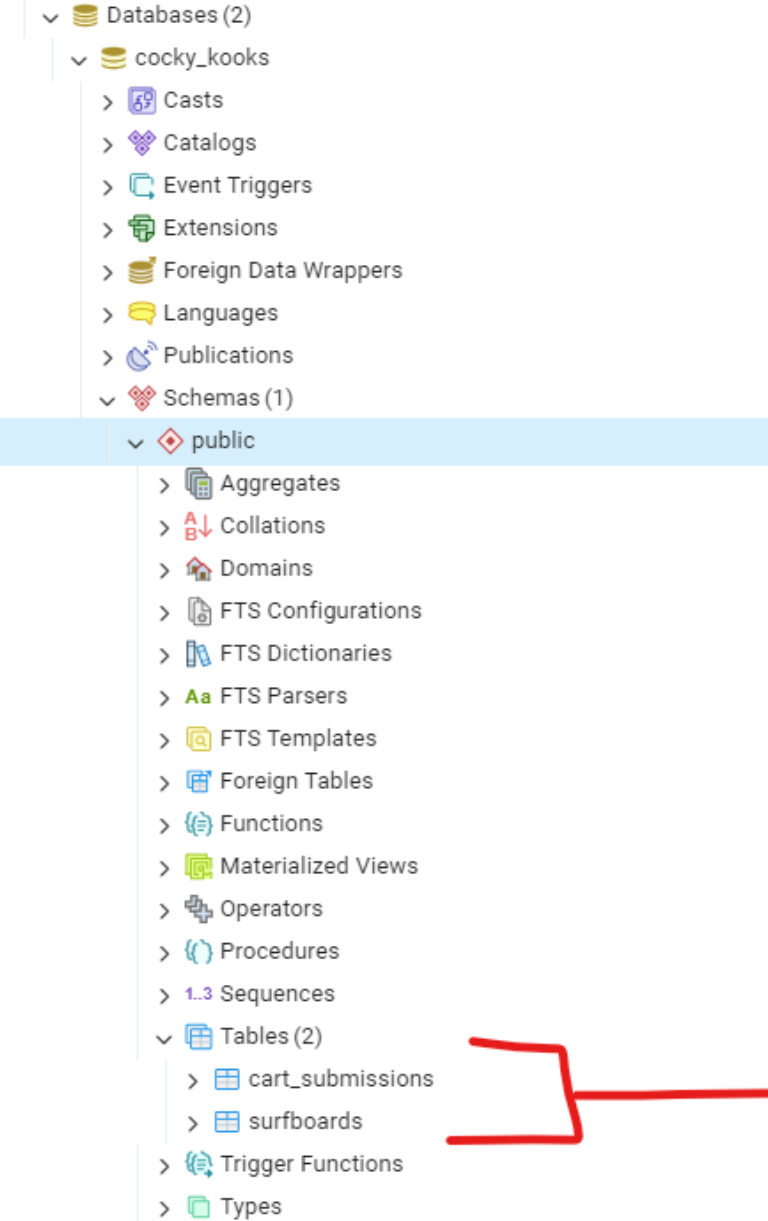


You should now see these two table created.

First right click on this Schemas folder, and hit refresh.



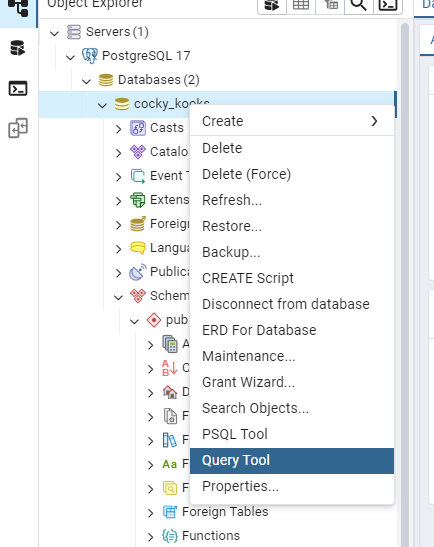
Then confirm that these two tables are there.



## Seed the Database

We’re now going to add data into the data base.

Right click the database, and launch the Query Tool again.



Copy and paste this code into the new window:

INSERT INTO surfboards (name, price, description, image\_url) VALUES

('The Barrel Chaser', 599.99, 'Perfect for fast, hollow waves.', '/images/board-1.jpg'),

('Lazy Daze', 499.99, 'Great for small mellow surf.', '/images/board-2.jpg'),

('Wave Slayer', 649.99, 'Lightweight high-performance shortboard.', '/images/board-3.jpg'),

('Sunset Cruiser', 699.99, 'Longboard designed for smooth rides.', '/images/board-4.jpg'),

('Pocket Rocket', 629.99, 'Compact, aggressive board for tight turns.', '/images/board-5.jpg'),

('The Classic', 749.99, 'Noserider with timeless design.', '/images/board-6.jpg'),

('Big Kahuna', 799.99, 'Massive longboard for big guys.', '/images/board-7.jpg'),

('Mini Mal', 599.99, 'Mid-length board for easy paddling.', '/images/board-8.jpg'),

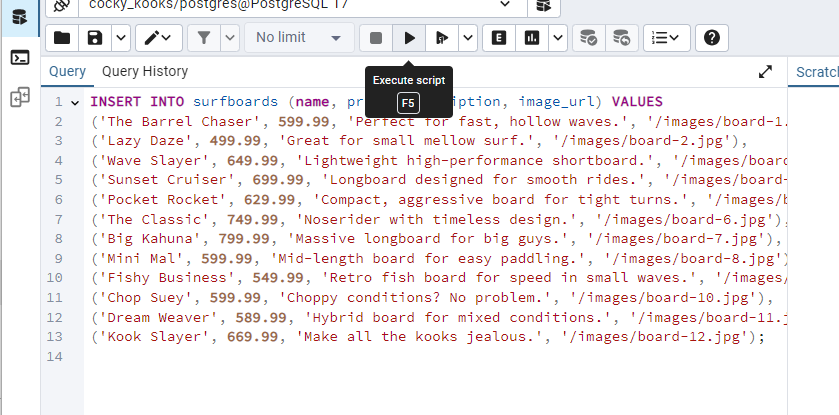
('Fishy Business', 549.99, 'Retro fish board for speed in small waves.', '/images/board-9.jpg'),

('Chop Suey', 599.99, 'Choppy conditions? No problem.', '/images/board-10.jpg'),

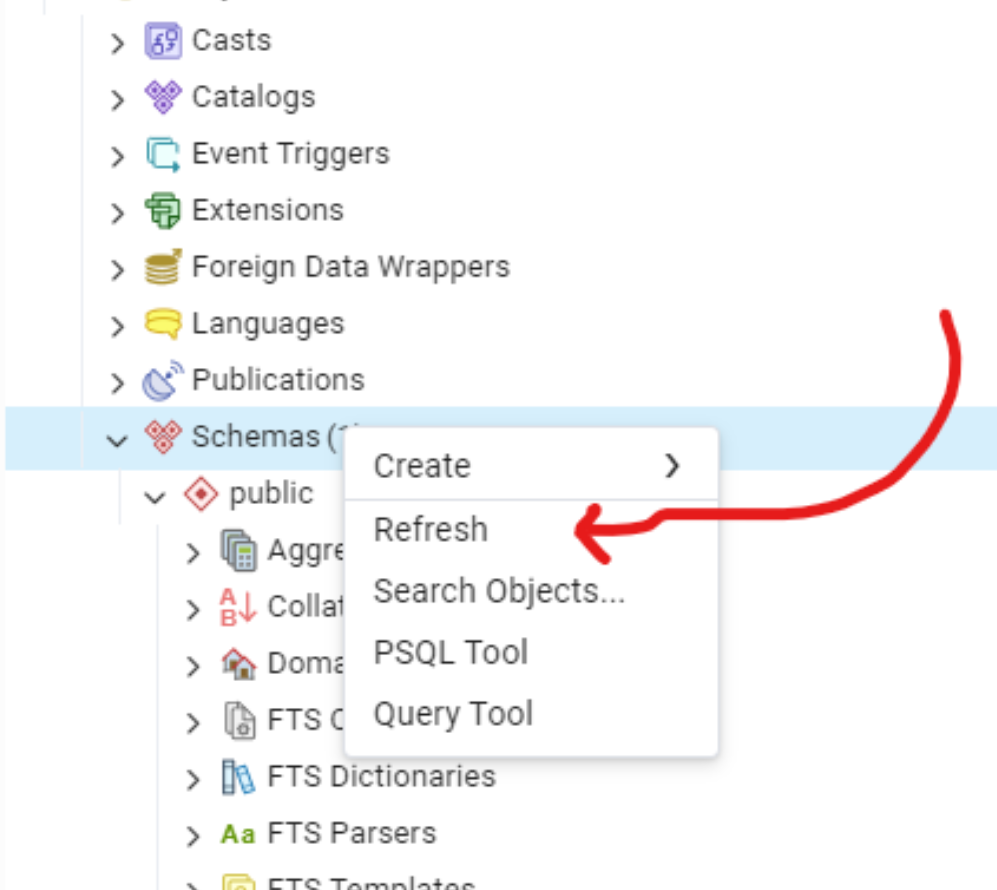
('Dream Weaver', 589.99, 'Hybrid board for mixed conditions.', '/images/board-11.jpg'),

('Kook Slayer', 669.99, 'Make all the kooks jealous.', '/images/board-12.jpg');

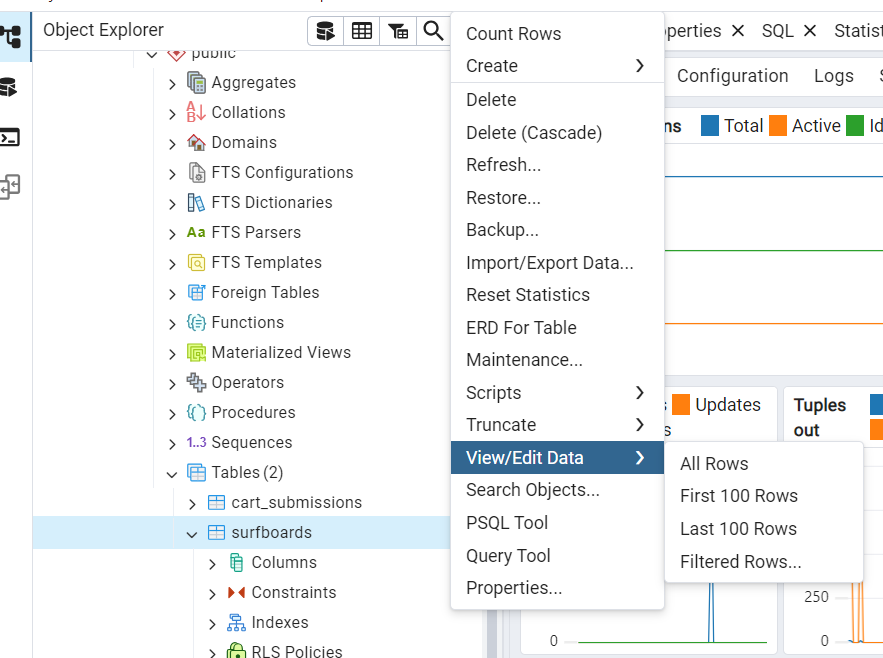
Hit the execute button:

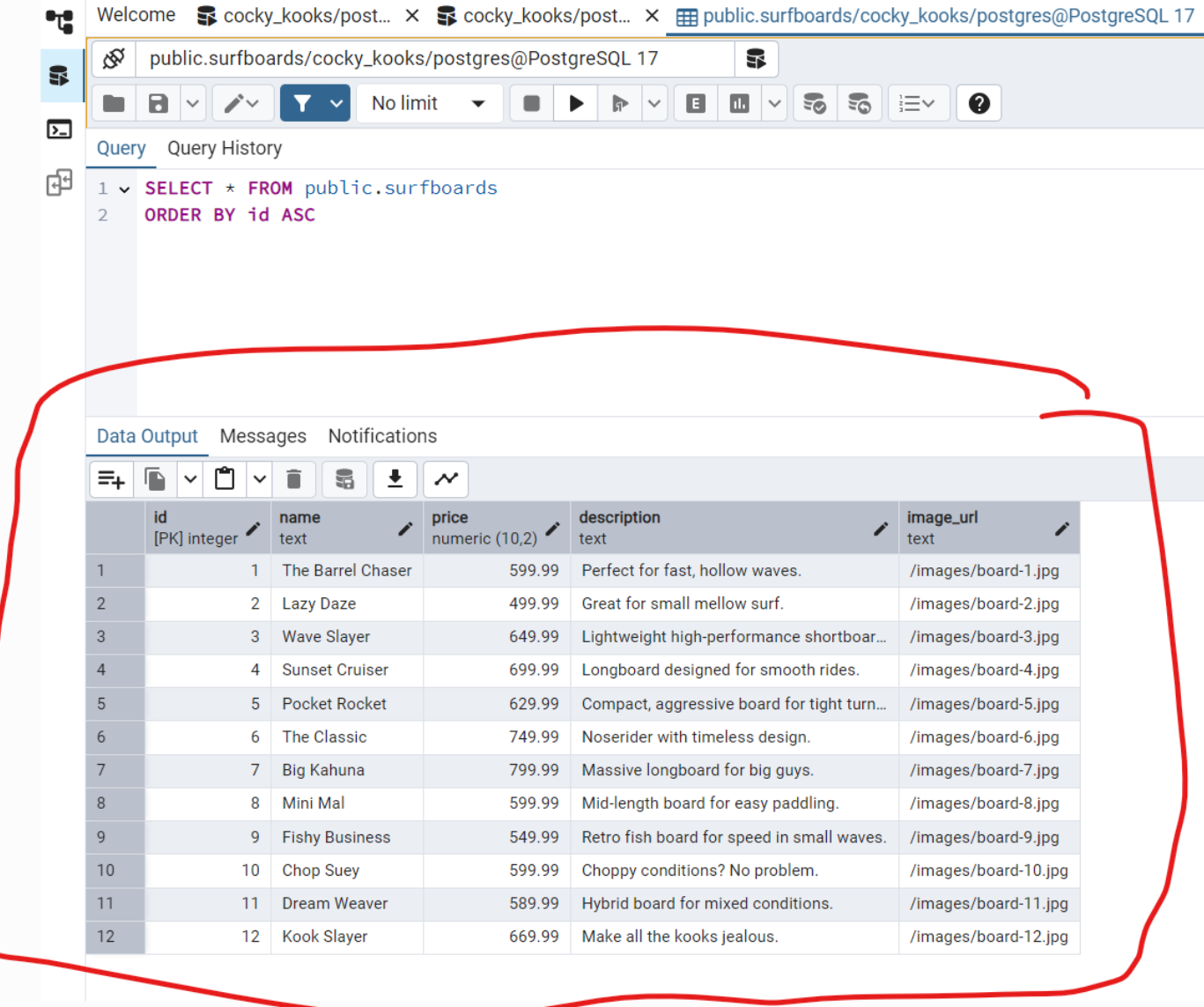


Refresh again:



Now, right click the surfboards table and click to the View/Edit Data > All Rows.



You’ll see these entries added into the table:  


**Extra bonus**, this code is what is used to show you the entries in the table:

SELECT \* FROM public.surfboards

ORDER BY id ASC

# Backend Setup and Development

## Install Node:

<https://nodejs.org/en>

Go there, download it, install it.

Node is the backend code that facilitates the communications between the frontend, backend, and database.

**Frontend** code runs on the end user’s compter.

**Backend** code runs on a server in a remote location. Meaning, not the code running on the end user’s computer.

The **database** is also in a remote location, not on the end user’s computer.

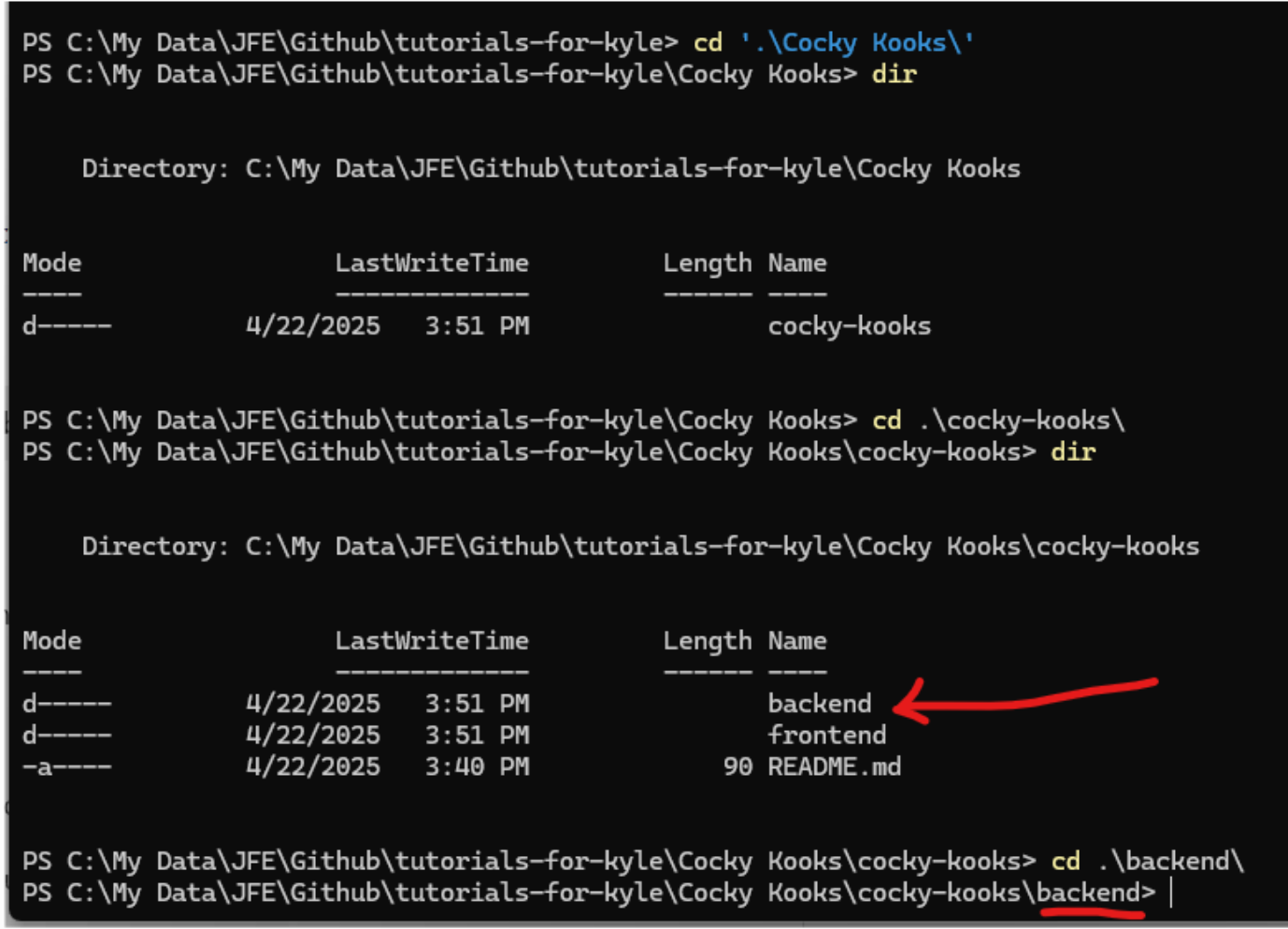
## Setup the backend project

Launch a command prompt. On Windows you can use PowerShell, on Mac use bash.

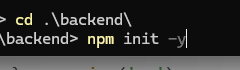
Navigate to the folder where you unzipped your code.

To navigate, you use the “cd” command, which stands for change directory.

Navigate to the folder named “backend”. Like this:



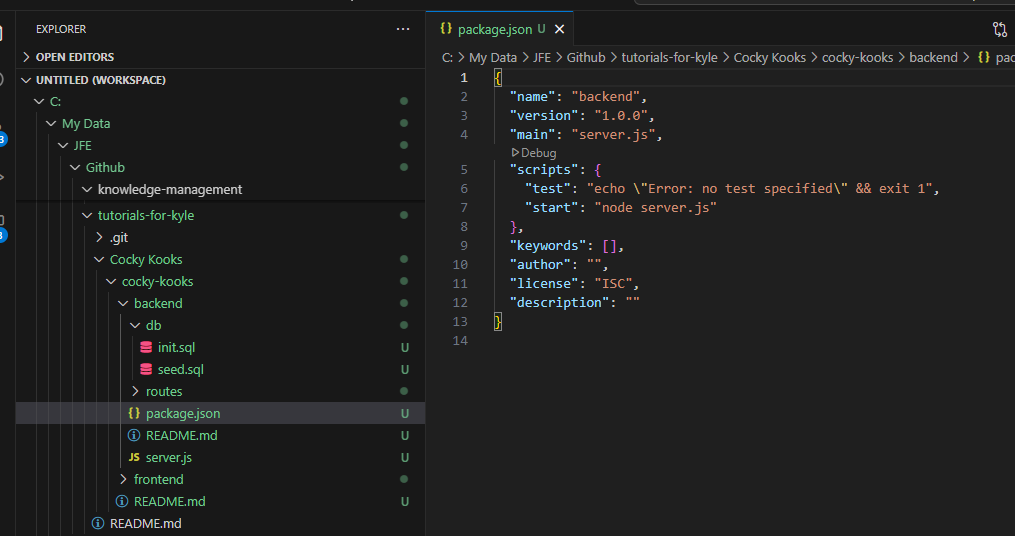
Initialize the backend project, by typing this:



so … once you’re at the backend folder location, type: **npm init -y**

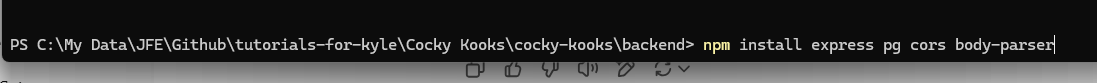
After you run that command, a new file will be created named package.json

Here’s a preview of the file that gets created.



Next, we’ll install the dependencies for the backend project.

Go back to the command prompt, while you’re at that same backend folder location, enter this:  
npm install express pg cors body-parser



Hit enter.

Now, all of the dependencies that you need for the backend project will be installed.

## Starter Code for the Backend Server

Paste the code below into this file: backend/server.js

const express = require('express');

const cors = require('cors');

const bodyParser = require('body-parser');

const { Pool } = require('pg');

const boardsRoutes = require('./routes/boards');

const submitRoutes = require('./routes/submit');

const app = express();

const port = 3001;

app.use(cors());

app.use(bodyParser.json());

// Database pool

const pool = new Pool({

user: 'your\_username',

host: 'localhost',

database: 'cocky\_kooks',

password: 'your\_password',

port: 5432,

});

app.use((req, res, next) => {

req.db = pool;

next();

});

// Routes

app.use('/api/boards', boardsRoutes);

app.use('/api/submit', submitRoutes);

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

## API Code for the Backend Server

Paste the code below into this file: backend/routes/boards.js

const express = require('express');

const router = express.Router();

router.get('/', async (req, res) => {

try {

const result = await req.db.query('SELECT \* FROM surfboards');

res.json(result.rows);

} catch (error) {

console.error(error);

res.status(500).send('Server Error');

}

});

module.exports = router;

Then paste the code below into this file: backend/routes/submit.js

const express = require('express');

const router = express.Router();

router.post('/', async (req, res) => {

const { name, email, message, selectedBoards } = req.body;

try {

await req.db.query(

'INSERT INTO cart\_submissions (name, email, message, selected\_boards) VALUES ($1, $2, $3, $4)',

[name, email, message, JSON.stringify(selectedBoards)]

);

res.json({ message: "Thank you, we've received your request and will be in touch." });

} catch (error) {

console.error(error);

res.status(500).send('Server Error');

}

});

module.exports = router;

# Frontend Development

## Main surfboard catalog page

In the root of the frontend project in the index.html file, paste this following code.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Cocky Kooks</title>

<link rel="stylesheet" href="css/styles.css">

</head>

<body>

<header>

<div id="cart-banner">Cart: <span id="cart-count">0</span> boards</div>

</header>

<main>

<div id="surfboards-grid"></div>

</main>

<script src="js/app.js"></script>

</body>

</html>

## CSS code

In: frontend/css/styles.css, paste this following code:

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

header {

background-color: #333;

color: white;

padding: 10px;

text-align: center;

cursor: pointer;

}

#surfboards-grid {

display: grid;

grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));

gap: 16px;

padding: 20px;

}

.surfboard-card {

border: 1px solid #ccc;

padding: 10px;

text-align: center;

}

.surfboard-card img {

width: 100%;

height: auto;

}

.selected {

border: 2px solid blue;

}

button {

margin-top: 10px;

padding: 8px 12px;

cursor: pointer;

}

## JavaScript Code

Add the following code to this file: js/app.js

let cart = [];

async function fetchSurfboards() {

const response = await fetch('http://localhost:3001/api/boards');

const boards = await response.json();

renderSurfboards(boards);

}

function renderSurfboards(boards) {

const grid = document.getElementById('surfboards-grid');

grid.innerHTML = '';

boards.forEach(board => {

const card = document.createElement('div');

card.className = 'surfboard-card';

card.dataset.id = board.id;

const img = document.createElement('img');

img.src = board.image\_url;

img.alt = board.name;

const name = document.createElement('h3');

name.textContent = board.name;

const description = document.createElement('p');

description.textContent = board.description;

const price = document.createElement('p');

price.textContent = `$${board.price}`;

const button = document.createElement('button');

button.textContent = 'Add to Cart';

button.addEventListener('click', () => toggleCart(board, card, button));

card.appendChild(img);

card.appendChild(name);

card.appendChild(description);

card.appendChild(price);

card.appendChild(button);

grid.appendChild(card);

});

}

function toggleCart(board, card, button) {

const index = cart.findIndex(item => item.id === board.id);

if (index === -1) {

cart.push(board);

card.classList.add('selected');

button.textContent = 'Remove from Cart';

} else {

cart.splice(index, 1);

card.classList.remove('selected');

button.textContent = 'Add to Cart';

}

updateCartCount();

}

function updateCartCount() {

document.getElementById('cart-count').textContent = cart.length;

localStorage.setItem('cart', JSON.stringify(cart));

}

document.getElementById('cart-banner').addEventListener('click', () => {

window.location.href = 'cart.html';

});

// Load existing cart from localStorage

window.addEventListener('load', () => {

const storedCart = localStorage.getItem('cart');

if (storedCart) {

cart = JSON.parse(storedCart);

updateCartCount();

}

fetchSurfboards();

});

## Cart Code

### HTML Code

In the cart.html, paste the following code.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Your Cart - Cocky Kooks</title>

<link rel="stylesheet" href="css/styles.css">

</head>

<body>

<header>

<div id="home-banner">← Back to Surfboards</div>

</header>

<main id="cart-page">

<div id="cart-list"></div>

<div id="checkout-form">

<h2>Checkout</h2>

<form id="submit-form">

<div>

<label>Name:</label><br>

<input type="text" id="name" required>

</div>

<div>

<label>Email:</label><br>

<input type="email" id="email" required>

</div>

<div>

<label>Message:</label><br>

<textarea id="message" rows="4">I would like to buy these boards.</textarea>

</div>

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

</form>

<div id="form-success" style="display:none; color: green; margin-top: 10px;"></div>

</div>

</main>

<script src="js/cart.js"></script>

</body>

</html>

### Update your CSS code a bit

Add this to the bottom of your styles.css

#cart-page {

display: flex;

padding: 20px;

gap: 20px;

}

#cart-list {

flex: 2;

}

#checkout-form {

flex: 1;

border-left: 1px solid #ccc;

padding-left: 20px;

}

form div {

margin-bottom: 10px;

}

### JavaScript Code

Add the following code to the cart.js file:

let cart = [];

function renderCart() {

const cartList = document.getElementById('cart-list');

cartList.innerHTML = '';

if (cart.length === 0) {

cartList.textContent = 'Your cart is empty.';

return;

}

cart.forEach(board => {

const card = document.createElement('div');

card.className = 'surfboard-card';

const img = document.createElement('img');

img.src = board.image\_url;

img.alt = board.name;

const name = document.createElement('h3');

name.textContent = board.name;

const description = document.createElement('p');

description.textContent = board.description;

const price = document.createElement('p');

price.textContent = `$${board.price}`;

card.appendChild(img);

card.appendChild(name);

card.appendChild(description);

card.appendChild(price);

cartList.appendChild(card);

});

}

async function submitForm(event) {

event.preventDefault();

const name = document.getElementById('name').value.trim();

const email = document.getElementById('email').value.trim();

const message = document.getElementById('message').value.trim();

if (!name || !email) {

alert('Name and Email are required.');

return;

}

try {

const response = await fetch('http://localhost:3001/api/submit', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({

name,

email,

message,

selectedBoards: cart

})

});

const result = await response.json();

document.getElementById('form-success').textContent = result.message;

document.getElementById('form-success').style.display = 'block';

// Clear cart

localStorage.removeItem('cart');

cart = [];

renderCart();

} catch (error) {

console.error('Error submitting form:', error);

alert('There was an error. Please try again.');

}

}

document.getElementById('submit-form').addEventListener('submit', submitForm);

// Home banner click

document.getElementById('home-banner').addEventListener('click', () => {

window.location.href = 'index.html';

});

// Load cart from localStorage

window.addEventListener('load', () => {

const storedCart = localStorage.getItem('cart');

if (storedCart) {

cart = JSON.parse(storedCart);

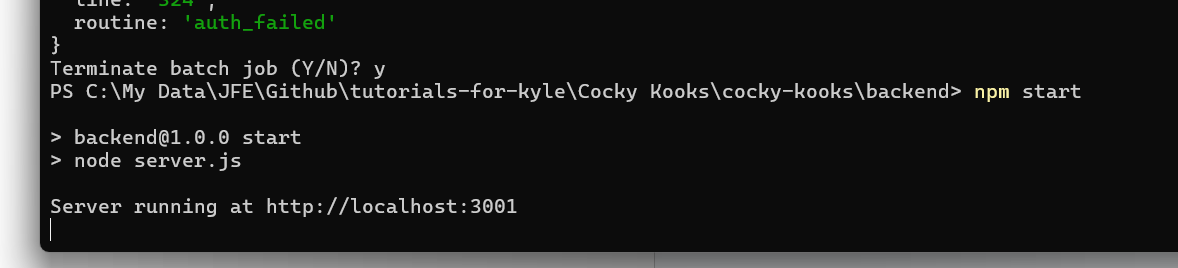
}

renderCart();

});

# Running the Projects

## Backend



Just type:

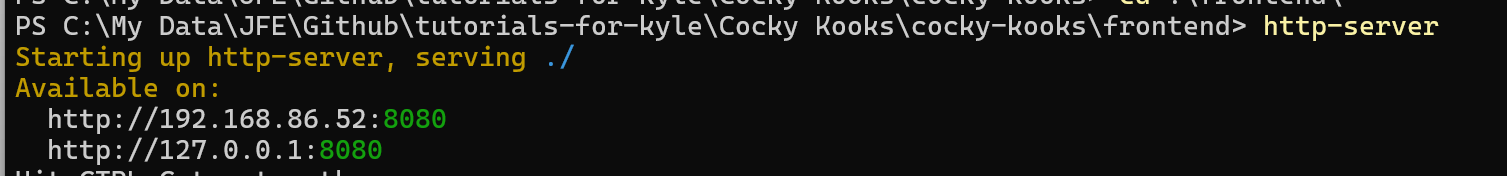
npm start

## Frontend

We’ll need to install this:  
<https://www.npmjs.com/package/http-server>

To install it, do this at the command line:  
npm i http-server

Here’s how you run it. You’ll be running from your “frontend” folder.

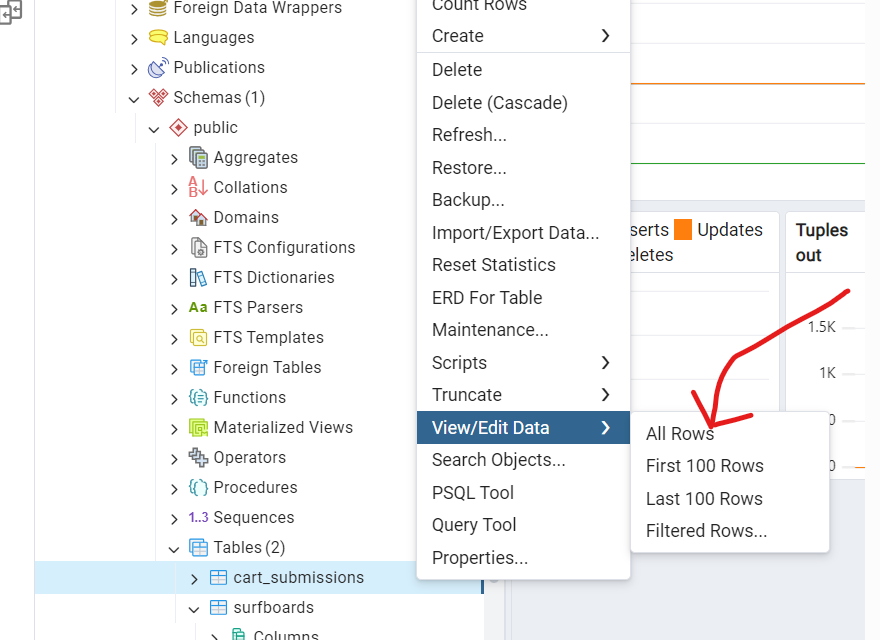


CD to the frontend folder, and type:

http-server

# Confirming Submissions

When the user makes their surfboard submission, confirm that the submissions are added to the database.



SELECT \* FROM cart\_submissions;

