Key Principles for This Challenge

- **Project First:** Every new concept is learned with the immediate goal of implementing it in a project.
- **Build & Iterate:** Don't aim for perfection on the first try. Get a working version done, then go back and improve it.
- **Show Your Work:** Use your HTML/CSS skills to make every project look good. Add them to a portfolio page as you complete them.
- **Stay Accountable:** Use the #100DaysOfCode hashtag on social media to share your daily progress.

Phase 1: JavaScript & DOM Mastery (Days 1-25)

Goal: Become proficient at making interactive, client-side web applications using vanilla JavaScript.

- **Days 1-5: JavaScript Fundamentals Review** 🧠
    - Quickly refresh core concepts: variables, data types, functions, arrays, objects, and loops.
    - **Your Task:** Solve 10-15 beginner algorithm challenges on a site like Edabit or freeCodeCamp to get your JS brain warmed up.
- **Days 6-15: Project 1 - Interactive Landing Page Components**
    - **Learn:** DOM manipulation (querySelector, addEventListener), and event handling.
    - **Build:** Enhance a landing page with several interactive components:
        - An image carousel/slider.
        - A "dark mode" toggle switch.
        - A modal/popup that appears on a button click.
        - An "FAQ" section with accordion-style collapsing answers.
- **Days 16-25: Project 2 - The "JavaScript30" Sprint** 🚀
    - **Task:** This is a high-intensity sprint. Complete the first 15 projects from the free JavaScript30 course by Wes Bos. These are small, fun projects (like a Drum Kit, a

CSS Clock, and a Type Ahead feature) that solidify your DOM manipulation skills in a practical way.

Phase 2: Working with APIs (Days 26-50)

Goal: Learn to fetch and display data from external sources, a core skill for modern web development.

- **Days 26-30: Learning Asynchronous JavaScript**
  - **Learn:** The concepts of Promises and the modern **async/await** syntax.
  - **Learn the fetch() API:** Practice making GET requests to free public APIs like the JSONPlaceholder to understand how to request and receive data.
- **Days 31-40: Project 3 - Recipe Finder App**
  - **Task:** Build an application that uses a free recipe API (like ThemealDB).
  - **Features:**
    - A search bar to look for recipes by name.
    - Display a list of results with images.
    - When a user clicks a result, display the full recipe details including ingredients and instructions.
- **Days 41-50: Project 4 - Personal Dashboard**
  - **Task:** Create a personalized browser dashboard that uses multiple APIs.
  - **Features:**
    - Display the current time.
    - Show the current weather for your location using the OpenWeatherMap API.
    - Display a new inspirational quote each day from an API.
    - Fetch and display a beautiful background image from an API like Unsplash.

Phase 3: Building a Backend (Days 51-75)

Goal: Create your own server and database to store and manage your application's data.

- **Days 51-60: Learning Node.js, Express, and REST APIs** 📦
    - **Learn:** Set up a server with **Node.js** and the **Express.js** framework.
    - **Learn:** Understand the principles of a **RESTful API** and how to create routes for GET, POST, PUT, and DELETE requests. Test your routes with a tool like Postman.
- **Days 61-65: Learning Databases with MongoDB** 🗄️
    - **Learn:** Set up a cloud database with **MongoDB Atlas**.
    - **Learn Mongoose:** Use Mongoose to create a schema and model for your data and connect it to your Express app.
- **Days 66-75: Project 5 - API for a To-Do List**
    - **Task:** Build a complete backend API. Don't build a front-end yet!
    - **Features:** The API should have endpoints to:
        - GET /tasks: Retrieve all tasks.
        - POST /tasks: Create a new task.
        - PUT /tasks/:id: Update an existing task (e.g., mark as complete).
        - DELETE /tasks/:id: Delete a task.
    - Ensure all data is saved, updated, and deleted from your MongoDB database.

Phase 4: Full-Stack & Deployment (Days 76-100)

Goal: Combine your frontend and backend skills to build and deploy a complete, data-driven web application.

- **Days 76-80: Learning a Frontend Framework (React Basics)** ✨
    - **Learn:** While you can build with vanilla JS, modern apps use frameworks. Learn the fundamentals of **React**: components, props, and state (useState hook).
    - **Task:** Rebuild one of your earlier, simple projects (like the dark mode toggle or a counter) as a React component to understand the basics.
- **Days 81-95: Capstone Project 6 - Full-Stack Blog or Product Tracker** 🏆

    - **Task:** Build a full-stack MERN (MongoDB, Express, React, Node.js) application from scratch.
    - **Option A: Blogging Platform.** Features: Create new posts with a title and content, view all posts on the homepage, click to view a single post, and delete posts.
    - **Option B: Personal Inventory Tracker.** Features: Add items with a name, quantity, and description. View all items. Update quantities. Delete items from inventory.
    - This project will tie everything together: your React frontend will fetch data from your Express backend, which will talk to your MongoDB database.
- **Days 96-100: Deployment & Portfolio Polish** 🚀
    - **Task:** Deploy your applications for the world to see.
        - Deploy your backend API using a service like **Render**.
        - Deploy your frontend React apps using **Netlify** or **Vercel**.
    - **Finalize:** Update your main portfolio page with links to the live versions and GitHub repositories for all six major projects. Write excellent README files for each one. Congratulations!