Take 2

APIs 103

Recap

- Promise
- CRUD
- REST
- HTTP Verbs

Systems Design 101

- Which systems make up our backend?

- A HTTP server
- A database

- Previously have used `json-server`
- It provides an API and stores its data in a file: `db.json`
- Let's change *one* component. Let's build `json-server` ourselves

Routing

- Which routes must we serve to our application?
- CRUD!
- POST `/resources`
- GET `/resources`
- GET `/resource/<id>`
- PUT `/resource/<id>`
- DELETE `/resource/<id>`

Express.js / Package Management

No frills.

```
$ bun add express

# or

$ npm init
$ npm install express
```

Express.js / "Hello World"

```
1  import express from "express";
2
3  const app = express();
4  const port = 8080;
5
6  app.get("/", (req, res) => {
7    res.send("Hello World!");
8  });
9
10 app.listen(port, () => {
11   console.log(`Listening on port ${port}...`);
12 });
```

Express.js / MVP

```
1  import express from "express";
2  import db from "./db.json";
3
4  const app = express();
5  const port = 8080;
6
7  app.get("/users", (req, res) => {
8    res.json(db["users"]);
9  });
10
11 app.listen(port, () => {
```

Express.js / To Be Continued

- Implement CRUD for users
- Use `node:fs` or `import`
- Implement dynamic resources
- Put it online (Vercel or Netlify)