Take 2

APIs 102

Recap

HTTP requests from the terminal

- requestb.in - website to show you HTTP requests that you make
- `curl` -> "See URL"

```
# Make a HTTP request to
# https://en9okehos12j7.x.pipedream.net/

$ curl -H 'Accept: application/json' \
  https://en9okehos12j7.x.pipedream.net/

{"success":true}
```

REST

- One convention of doing APIs is called RESTful APIs
- All RESTful APIs work the same, they revolve around a *Resource* like
  - BlogPost, Comment, User, Image, Appointment
- They use paths, called *API endpoints* to address resources (by id) or *collections*
  - `/blog_posts`, `/comments/5`, `/appointments`, `/user/1/images/4`
- They use HTTP verbs to determine actions with resources
  - List all blog posts
  - Show comment with id 5
  - Add a new appointment

RESTful APIs use CRUD

Example resource: `BlogPost`

- Create a `BlogPost`
- Read a `BlogPost`
- Update a `BlogPost`
- Delete a `BlogPost`

CRUD & HTTP verbs ("methods")

- **C**reate -> HTTP POST
- **R**ead -> HTTP GET (default)
- **U**pdate -> HTTP PUT or PATCH
- **D**elete -> HTTP DELETE

Putting it all together

- Create a new blog post: `POST /blog_posts`
- Show all comments: `GET /comments`
- Show the comment with the id 5: `GET /comments/5`
- Update the comment with the id 5: `PUT /comments/5`
- Delete the comment with the id 5: `DELETE /comments/5`

Try it out, start a JSON server

https://github.com/typicode/json-server

```
$ npx json-server db.json
```

```
$ curl http://localhost:3000/posts
[
  {
    "id": "1",
    "title": "a title",
    "views": 100
  }
]
```

Adding a post

```
$ curl -X POST -H "Content-Type: application/json" \
  -d '{ "title": "new post" }' localhost:3000/posts

{
  "id": "d64d",
  "title": "new post"
}
```

```
$ curl http://localhost:3000/posts

[
  {
    "id": "1",
    "title": "a title",
    "views": 100
  },
  {
    "id": "d64d",
    "title": "new post"
  }
]
```

Using verbs with fetch

```
const response = await fetch("https://example.org/post", {
  method: "POST",
  // ...
});
```

- Use json server to build a contacts API.
- Adapt the `db.json` to contain 'contacts' instead of 'posts' etc.
- Build a small app to list all contacts and add a new one.