brief git clarification / review

Why do we keep doing

```
git reset --hard
```

and

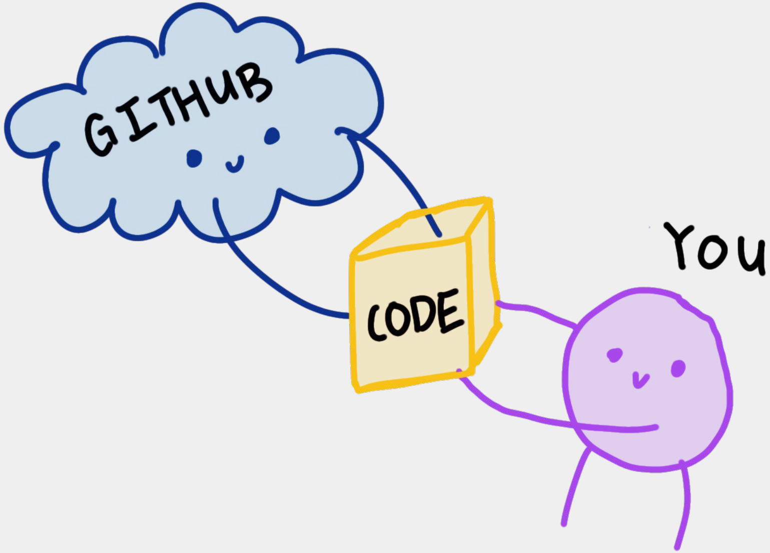
```
git checkout wX-stepY
```

like… what do these commands even do again…

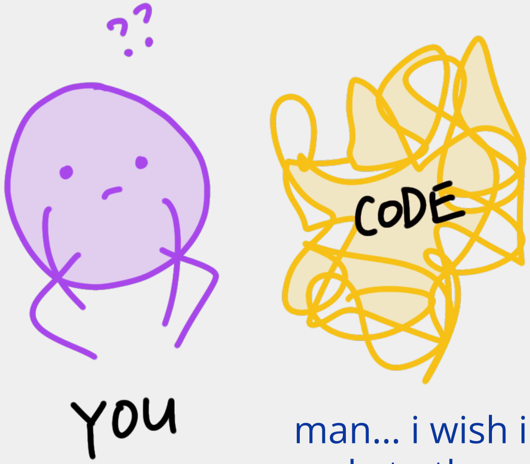
`git reset --hard`

resets the LOCAL version of your code (on your computer) to **match** the version that’s stored in the CLOUD (on github)

irreversibly deletes all local (tracked) changes

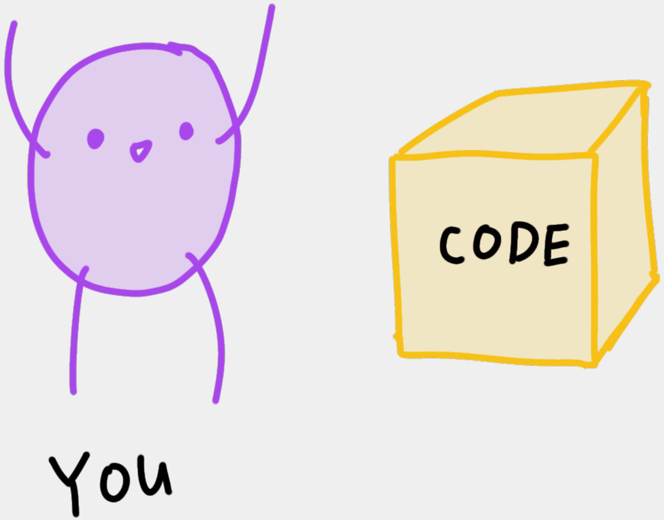

GITHUB
CODE
YOU

many minutes later...



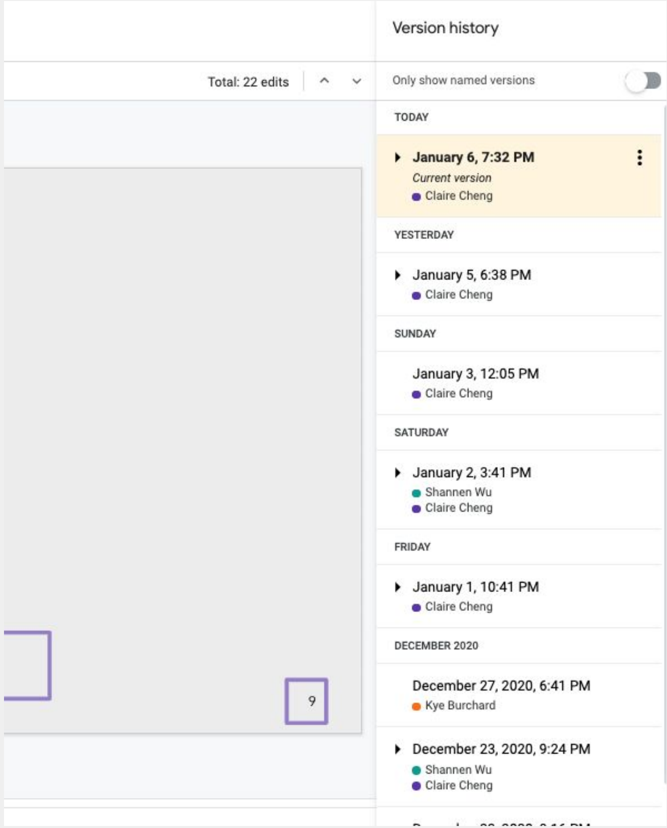
man... i wish i could just restore the code to the original working version but I'd have to press "undo" like 10,000 times and vscode won't even let me do that...

```
git reset --hard
```


You
CODE

```
git checkout wX-stepY
```

retrieves the *branch* “wX-stepY” from the cloud (github)



Version history

Total: 22 edits

Only show named versions

TODAY

January 6, 7:32 PM
Current version
● Claire Cheng

YESTERDAY

January 5, 6:38 PM
● Claire Cheng

SUNDAY

January 3, 12:05 PM
● Claire Cheng

SATURDAY

January 2, 3:41 PM
● Shannen Wu
● Claire Cheng

FRIDAY

January 1, 10:41 PM
● Claire Cheng

DECEMBER 2020

December 27, 2020, 6:41 PM
● Kye Burchard

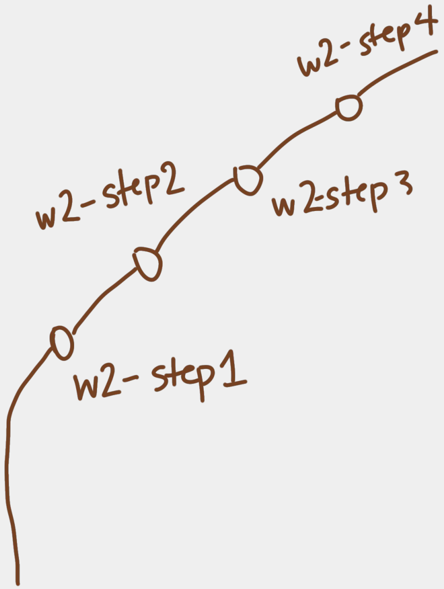
December 23, 2020, 9:24 PM
● Shannen Wu
● Claire Cheng

going through version history of a google doc and selecting a specific version to “restore”

→ it’s the same idea!

we want to "check out" a particular version of the code.

(in this case, it's an individual step of our workshop.)



hope that clarifies things a bit!

if you have more questions about git / version control, feel free to come to OH :)

Workshop 3 - More React, using APIs, and Routing

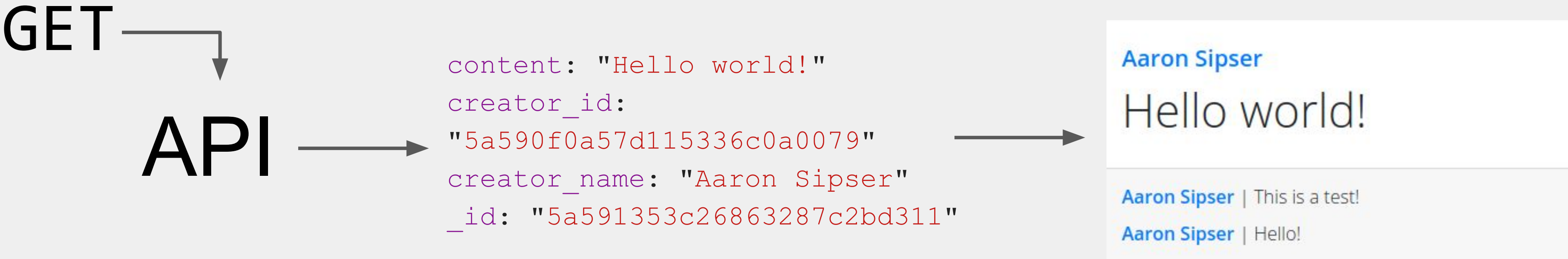
Claire Cheng + Helen Lu

Previously...

- From Workshop 2
  - Profile page in React
    - Components
    - State and Props
- From Lecture
  - What is an API?
  - HTTP requests
  - Asynchronous programming (briefly)

This Workshop!

- Back to Catbook!
  - Getting more familiar with React
  - Interacting with an API from the frontend
    - GET
    - POST



GET

API

```
content: "Hello world!"
creator_id:
"5a590f0a57d115336c0a0079"
creator_name: "Aaron Sipser"
_id: "5a591353c26863287c2bd311"
```

Aaron Sipser

Hello world!

Aaron Sipser | This is a test!

Aaron Sipser | Hello!

Getting started

- Navigate to catbook-react repository
- `git reset --hard`
- `git checkout w3-starter`
- `npm install`

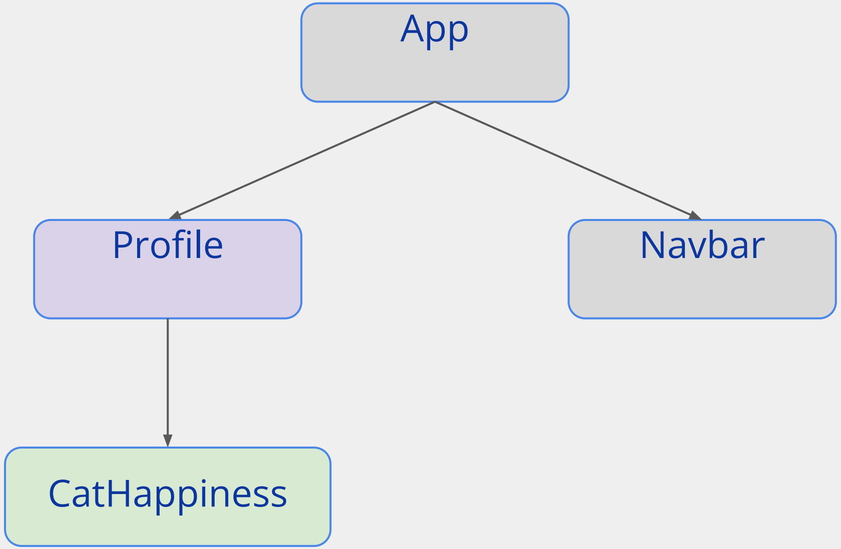
`weblab.to/questions` `<- general questions`

`weblab.to/q` `<- individual help`

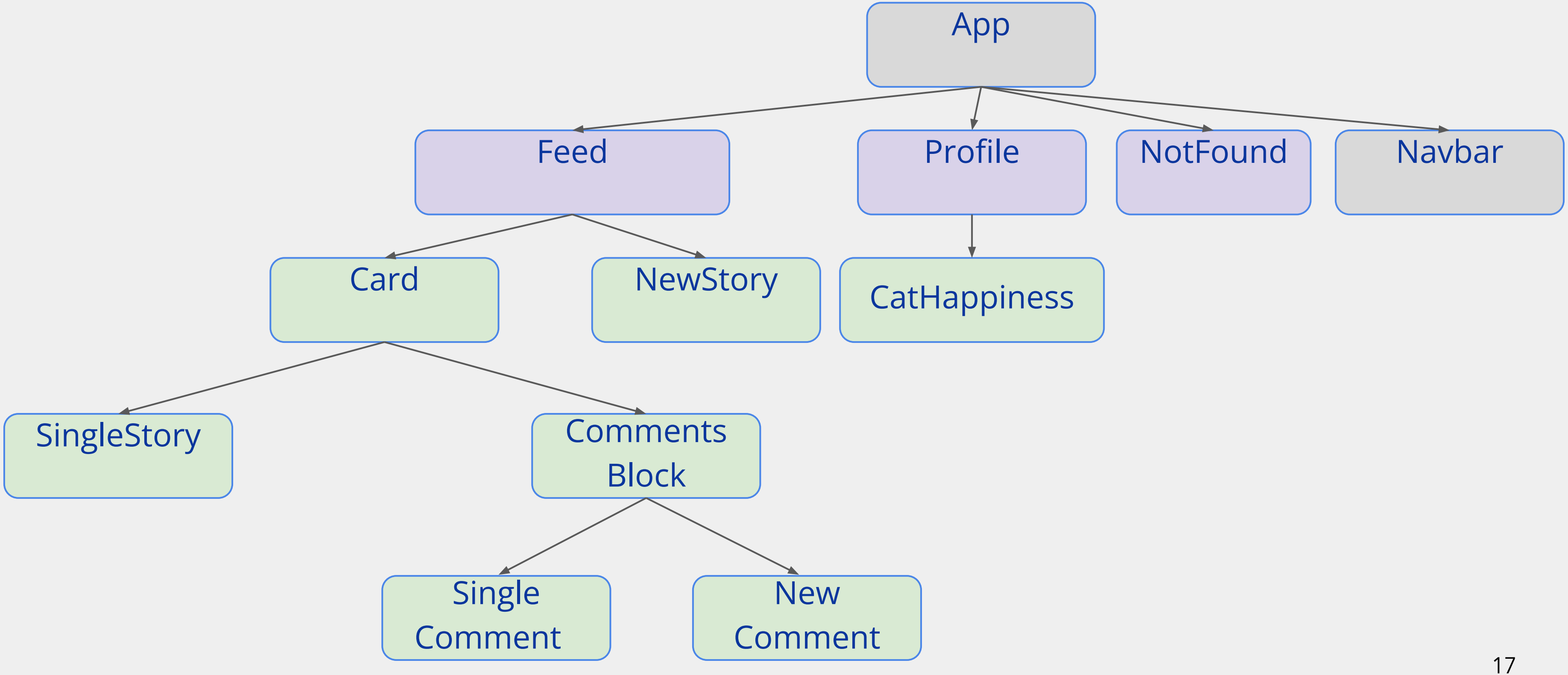
Overview

- Similar folder structure to last time
- New file!
    - `utilities.js`

Recall: component hierarchy before



Component hierarchy after today!



Goals for today

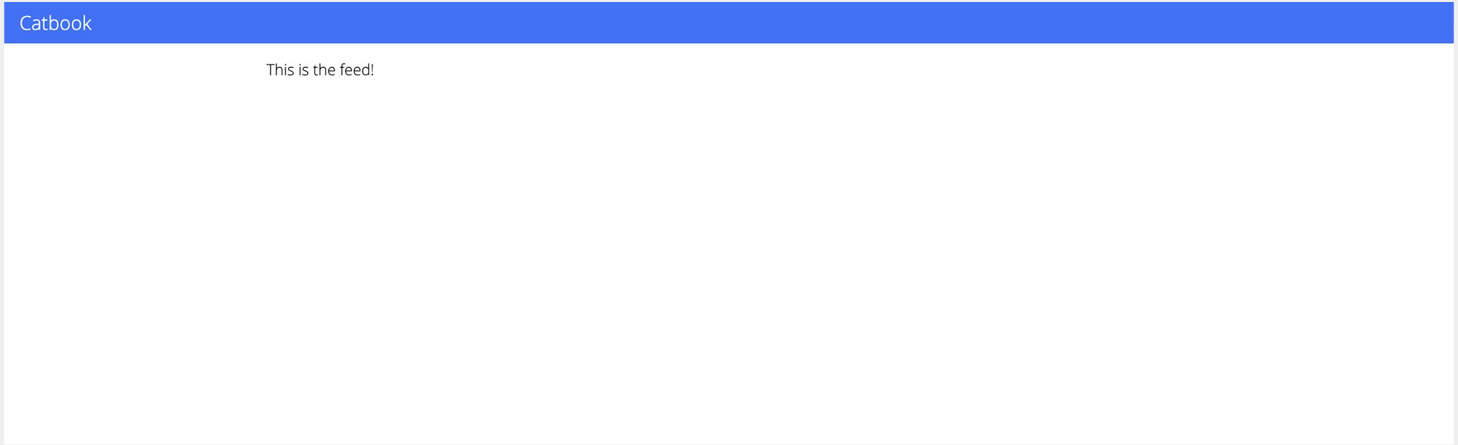
- The Feed page
  - Ability to see other posts
    - GET
  - Ability to make your own post
    - POST
- Frontend Routing

Let’s go!

- `npm run hotloader`
- In a browser, head to ‘`localhost:5000`’
- Should look a lot like workshop 2

Warmup

- In `App.js`
  - 1) Import `Feed.js`
  - 2) Render `Feed` instead of `Profile`



This is the feed!

Warmup

- Questions?
- Catch up
  - `git reset --hard`
  - `git checkout w3-step1`

Follow along: step 1

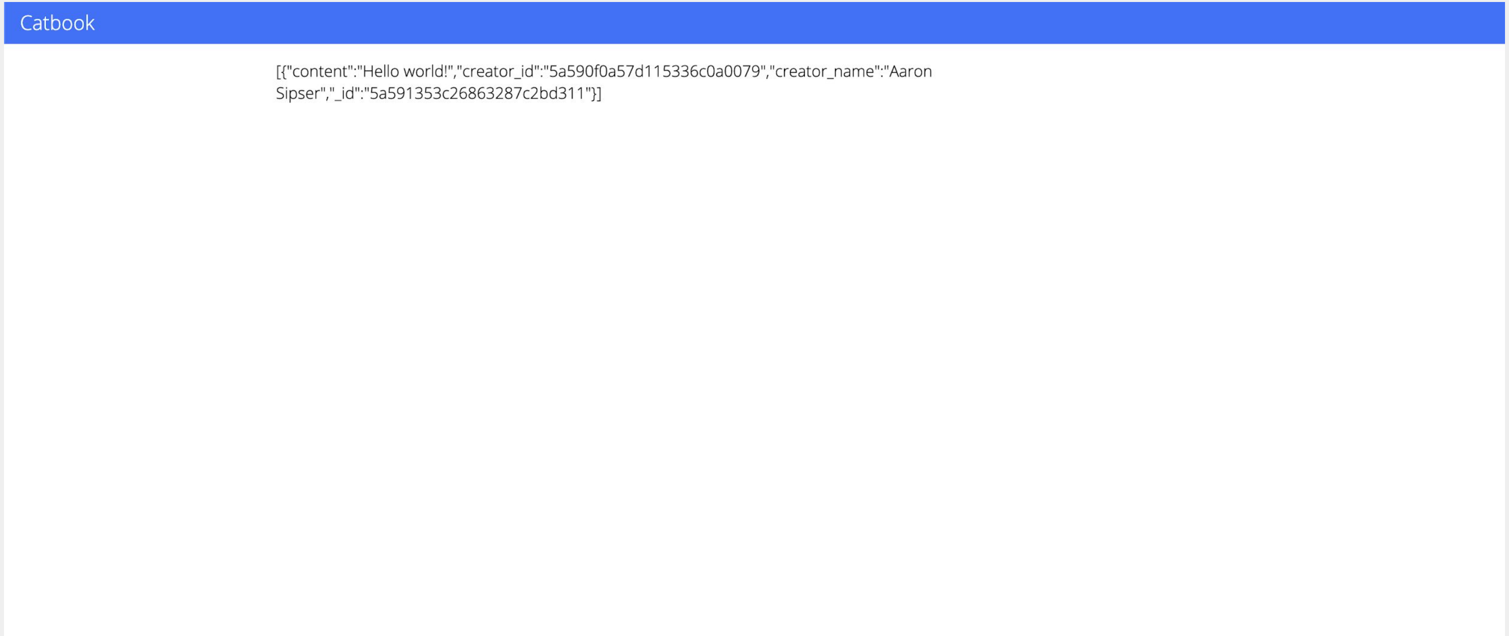
- In `Feed.js`
  - Let's get (GET) stories!
- Follow along with me
  - Hold stories in `state`
  - Get stories in `useEffect`
  - Show stories in `render`

`GET /api/stories`
Returns all stories
Parameters: none

`JSON.stringify(*list*)` makes a list render-able

Follow along: step 1

- Testing our GET



```
[{"content":"Hello world!","creator_id":"5a590f0a57d115336c0a0079","creator_name":"Aaron
Sipser","_id":"5a591353c26863287c2bd311"}]
```

Follow along: step 1

- Questions?
- Catch up
    - `git reset --hard`
    - `git checkout w3-step2`

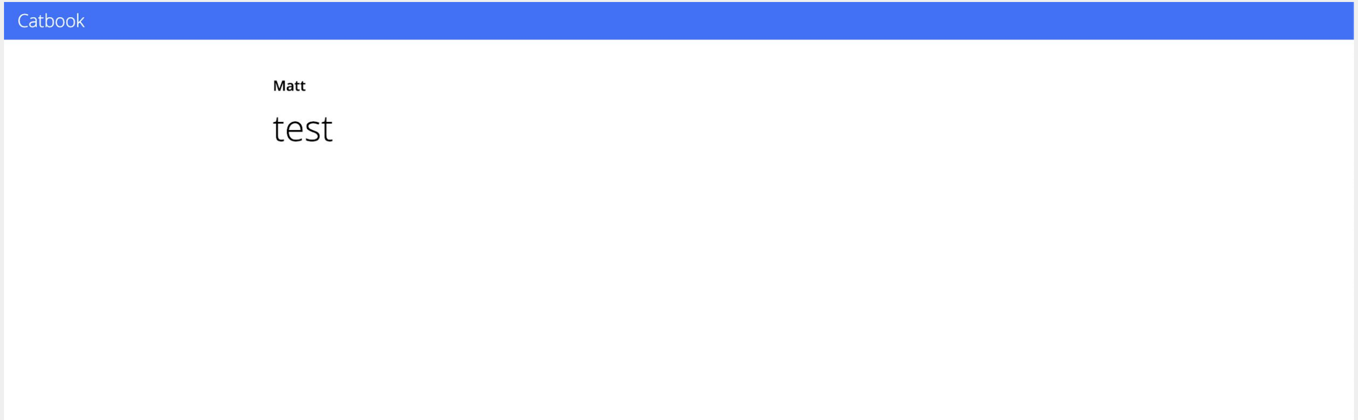
Your turn: step 2

- In `Feed.js`
    - Import `SingleStory.js`
    - Render a `SingleStory` component, passing in dummy name and content

Your turn: step 2

- In `SingleStory.js`
  - Let's make our stories prettier!
  - Use JSX to render passed in props

- CSS classes to use:
  - `Card-story`
  - `u-bold`
  - `Card-storyContent`



Matt

test

Your turn: step 2

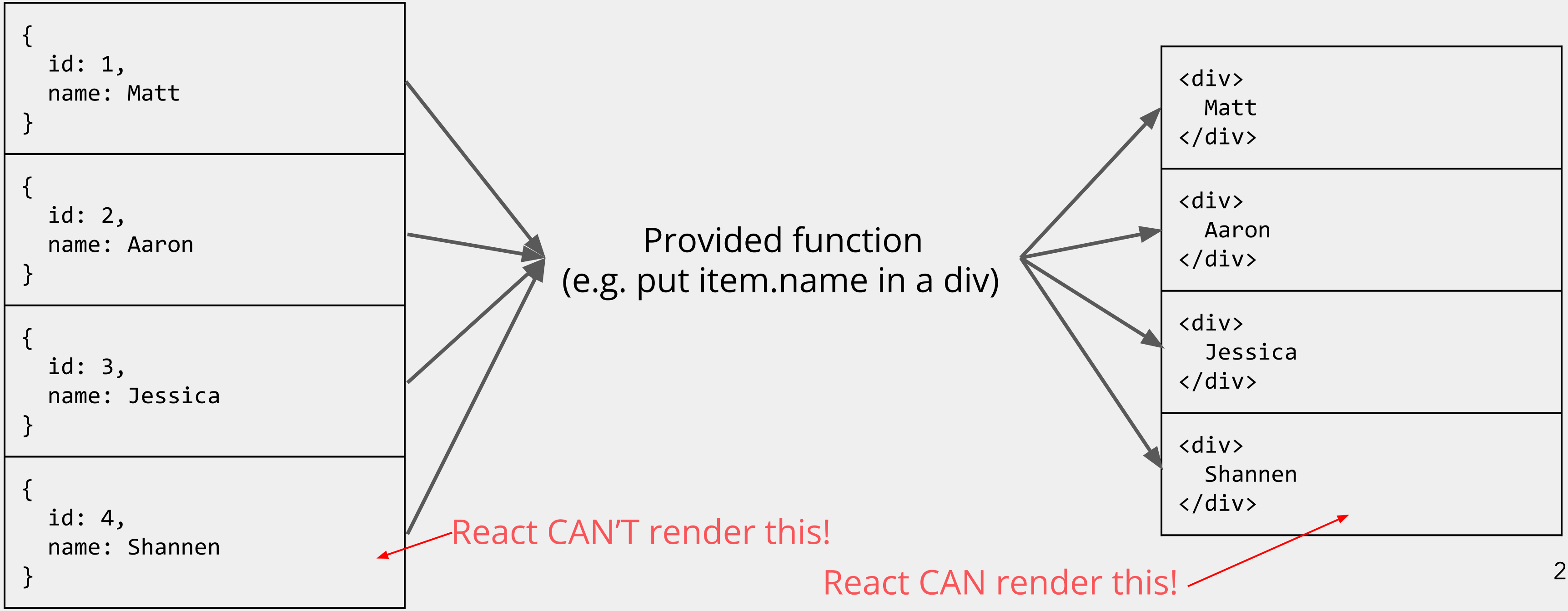
- Questions?
- Catch up
  - `git reset --hard`
  - `git checkout w3-step3`

Follow along: step 3

- In `Feed.js`
  - Putting our state and `SingleStory` together
  - Object array -> Component array?

Review: the map function

- Recall: the `map()` method creates a new array by applying a function to every element of the starting array.



```
{
  id: 1,
  name: Matt
}
```

```
{
  id: 2,
  name: Aaron
}
```

```
{
  id: 3,
  name: Jessica
}
```

```
{
  id: 4,
  name: Shannen
}
```

Provided function
(e.g. put item.name in a div)

```
<div>
  Matt
</div>
```

```
<div>
  Aaron
</div>
```

```
<div>
  Jessica
</div>
```

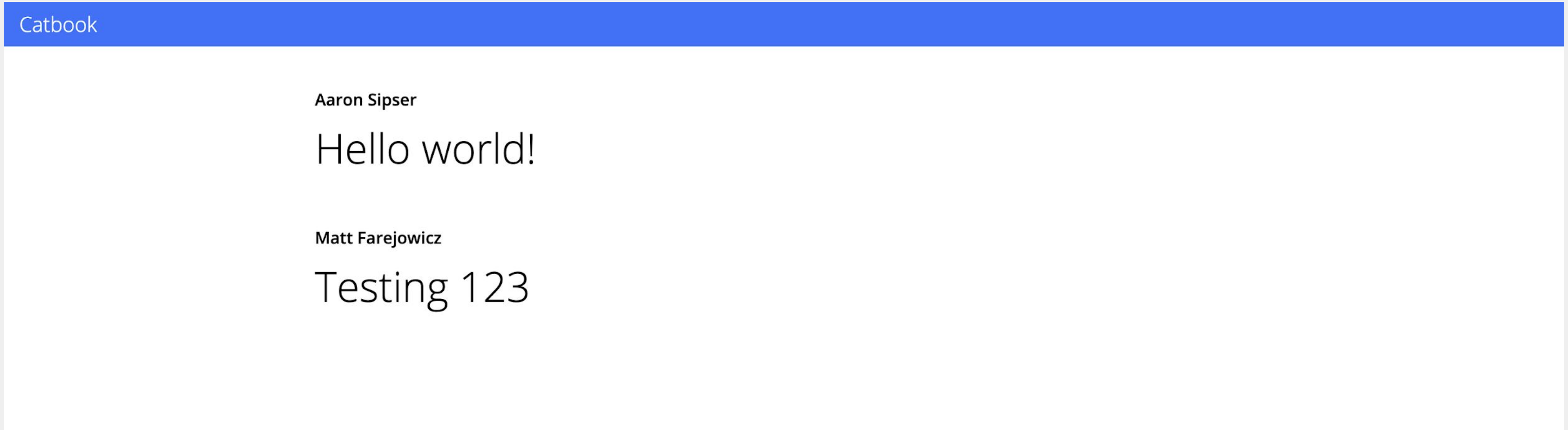
```
<div>
  Shannen
</div>
```

React CAN'T render this!

React CAN render this!

Follow along: step 3

- In `Feed.js`
  - Putting our state and `SingleStory` together
  - Object array -> Component array?



Catbook

Aaron Sipser

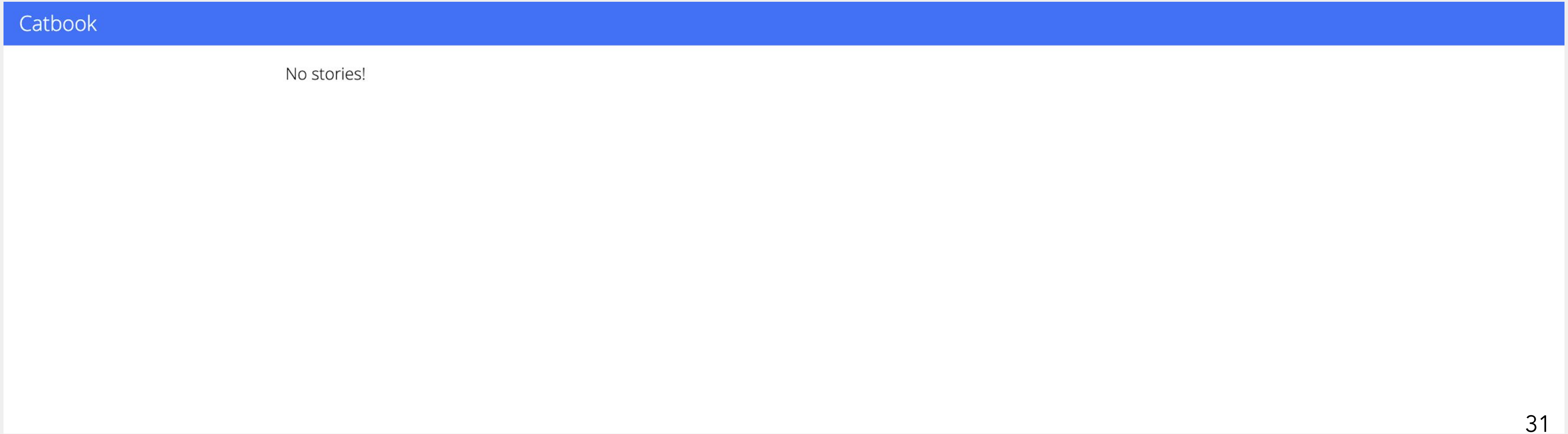
Hello world!

Matt Farejowicz

Testing 123

Follow along: step 3

- In `Feed.js`
  - What to do if no stories?



No stories!

Follow along: step 3

- Questions?
- Catch up
  - `git reset --hard`
  - `git checkout w3-step4`

That was GET, how about POST?

`POST /api/story`

Posts a new story

Parameters: `content` (string)

Follow along: step 4

- In `NewPostInput.js`
  - Let's make a component that can post (POST) stories

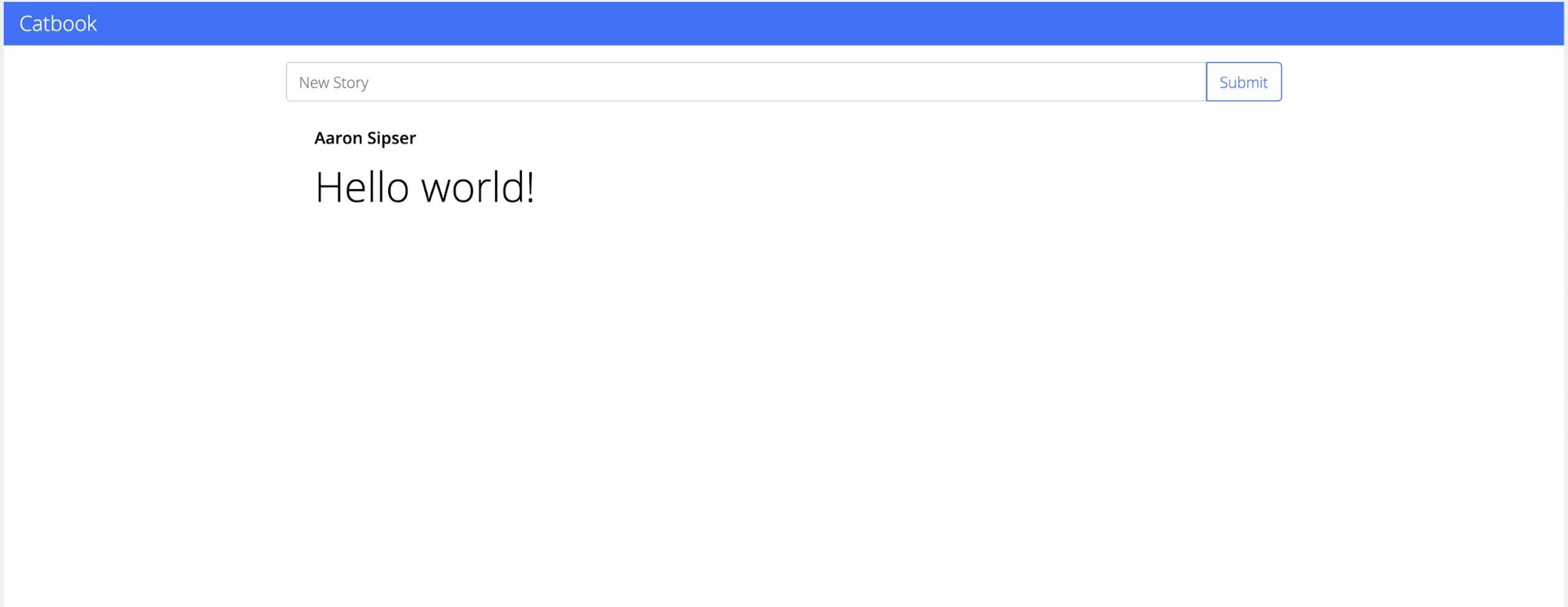
`POST /api/story`

Posts a new story

Parameters: `content` (string)

- Follow along with me
  - The `NewStory` component
  - Using it in `Feed`

Follow along: step 4



Aaron Sipser

Hello world!

Follow along: step 4

- Questions?
  - Please ask! We'll have you do something similar to steps 1-4 later!
- Catch up
  - `git reset --hard`
  - `git checkout w3-step5`

Homework:

React practice: weblab.to/homework3

Milestone 0: weblab.to/milestone0

let’s continue today :)

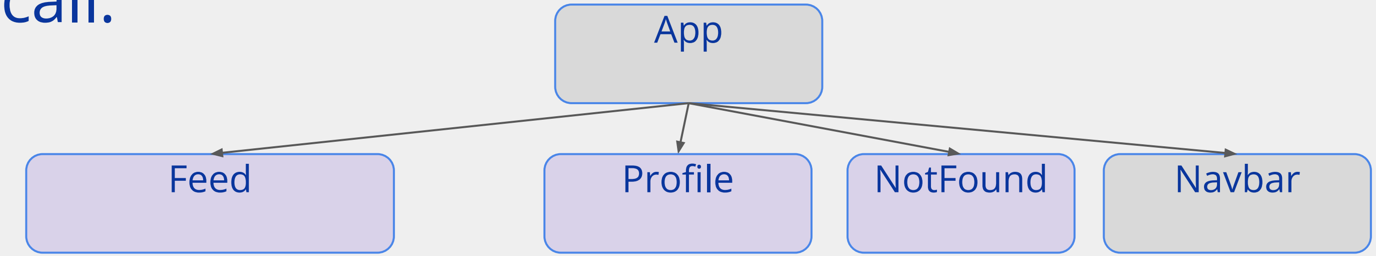
Follow along: step 5

If you haven’t done this already…

- `git reset --hard`
- `git checkout w3-step5`

App navigation?

- Recall:

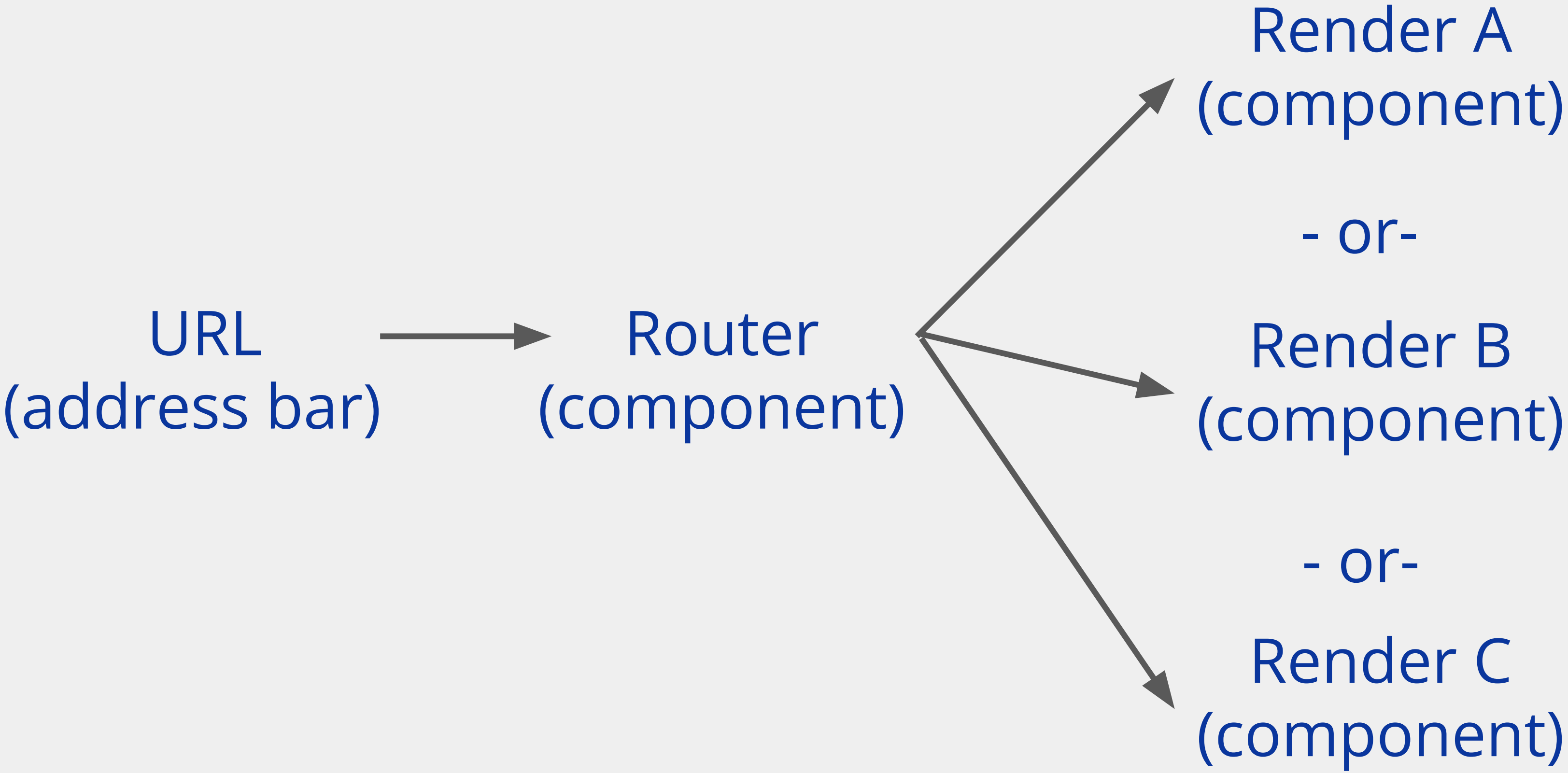


- Determine content via routing!
- Examples:
  - catbook.com
  - catbook.com/profile
  - catbook.com/asdf

App navigation?

- We will be using the Reach Router library
- https://reach.tech/router

App navigation?



URL
(address bar)

Router
(component)

Render A
(component)

- or-

Render B
(component)

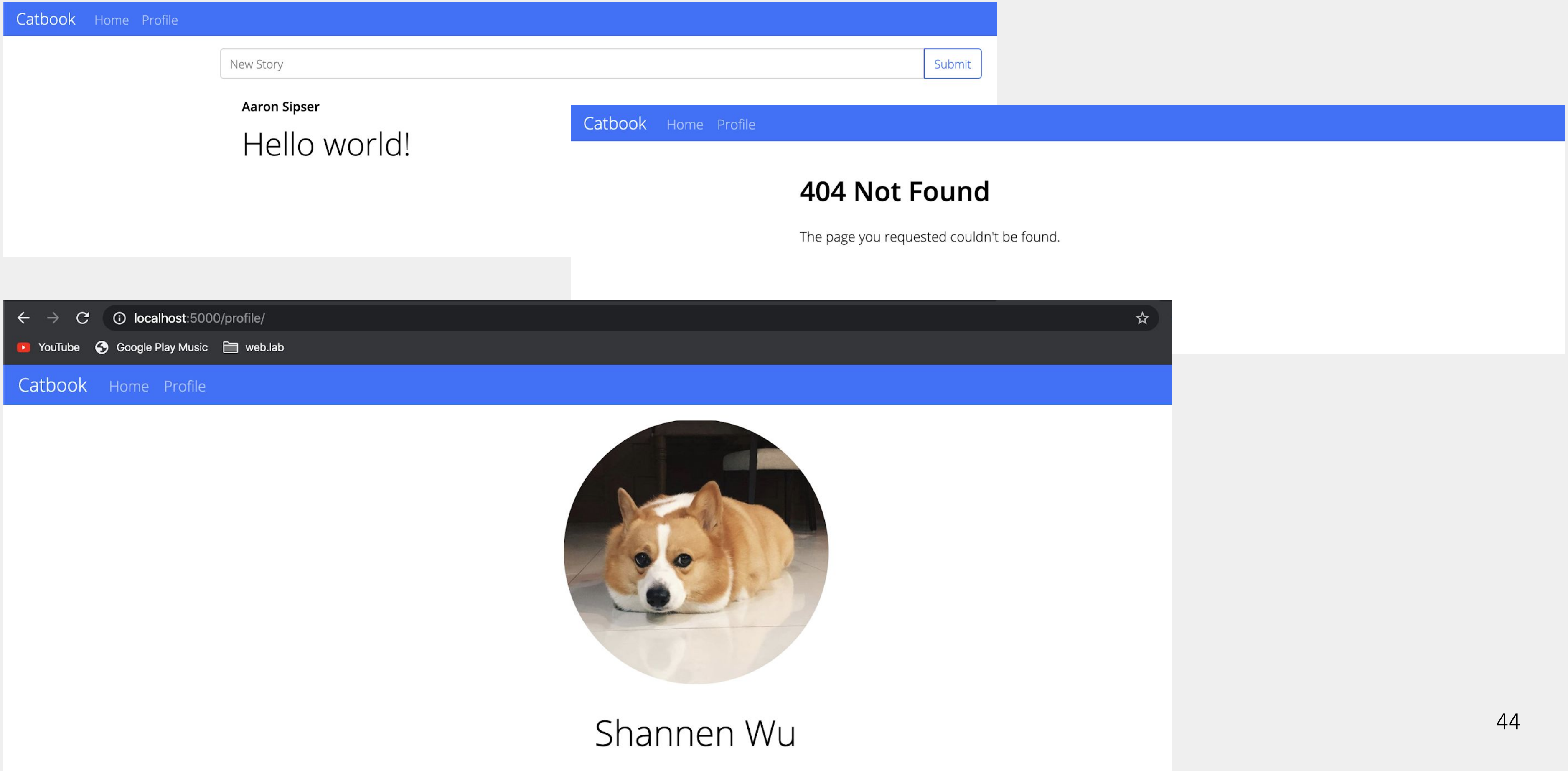
- or-

Render C
(component)

Follow along: step 5

- In `App.js`
  - Using the `Router` component
- In `NavBar.js`
  - Using the `Link` component

Follow along: step 5



Catbook Home Profile

New Story Submit

Aaron Sipser

Hello world!

Catbook Home Profile

404 Not Found

The page you requested couldn't be found.

localhost:5000/profile/

YouTube Google Play Music web.lab

Catbook Home Profile

Shannen Wu

Follow along: step 5

- Questions?
- Catch up
    - `git reset --hard`
    - `git checkout w3-step6`

Steps 6-8

- Now that stories are complete, it's time to work on comments.
- Steps 6-8 are highly analogous to steps 1-4, but we'll be working in `Card.js` instead of `Feed.js`
- Reminder of high level overview:
  - GET to obtain stories
  - `SingleComment` component renders data

Steps 6-8

Previously:

`GET /api/stories`

Returns all stories

Parameters: none

`POST /api/story`

Posts a new story

Parameters: `content`

Now:

`GET /api/comment`

Returns all comments for a story

Parameters: `parent` (string)

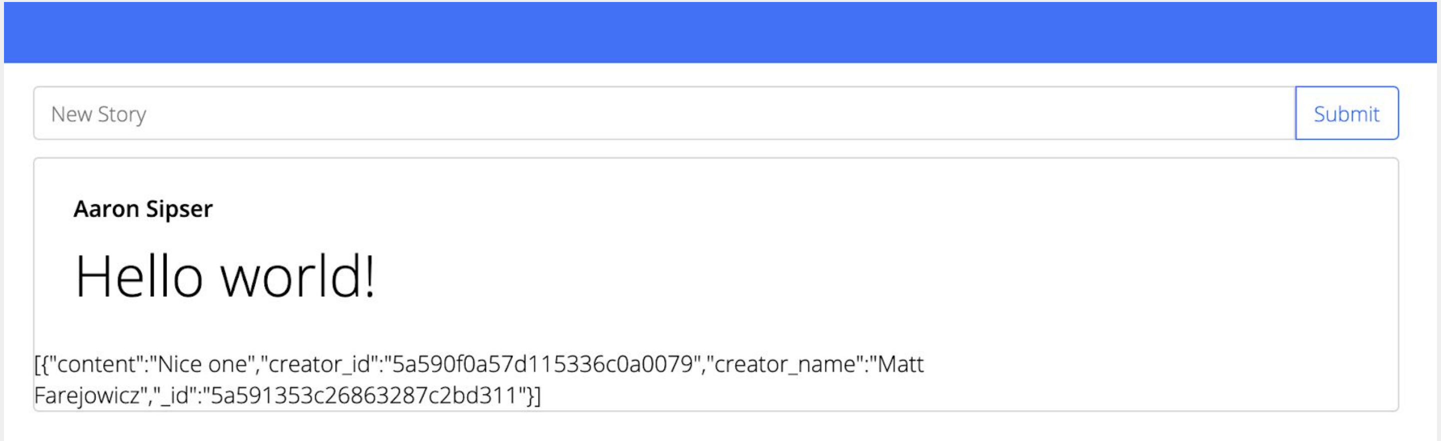
`POST /api/comment`

Posts a new comment

Parameters: `parent` (string)
`content`

Follow along: Step 6

- GET comments in `Card.js`
- Render them like stories
- Use `<Card (props) />` in `Feed.js`



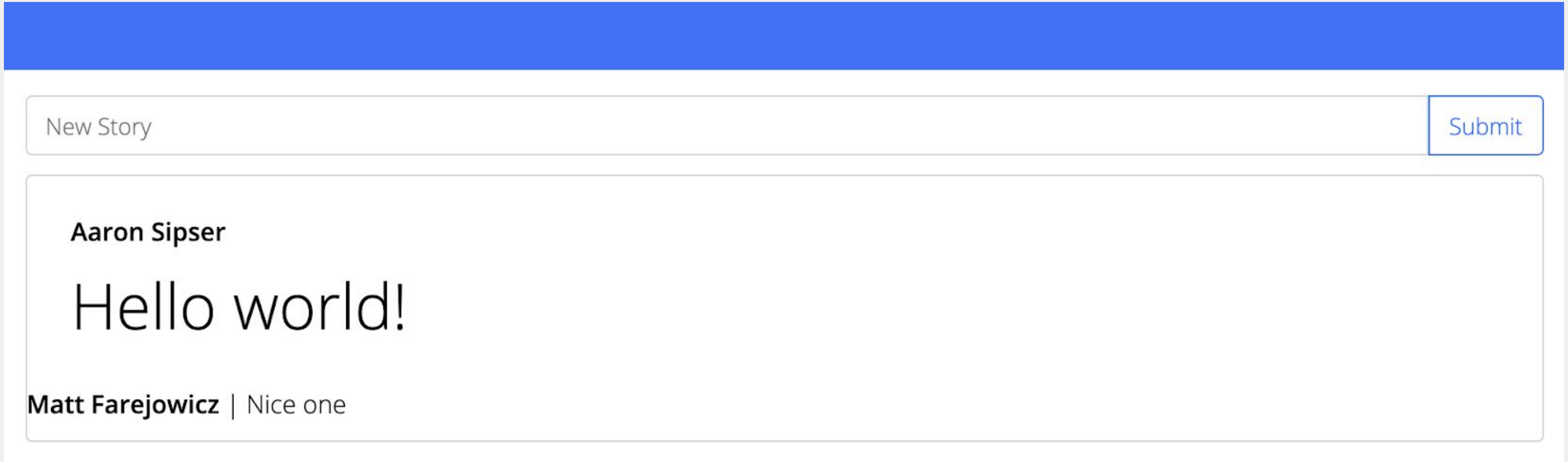
Aaron Sipser

Hello world!

```
[{"content":"Nice one","creator_id":"5a590f0a57d115336c0a0079","creator_name":"Matt
Farejowicz","_id":"5a591353c26863287c2bd311"}]
```

Your turn! Step 7

- Implement `SingleComment.js`
- Use `<SingleComment (props) />` in `Card.js` (like how we used <Card /> in Feed.js)



New Story

Submit

Aaron Sipser

Hello world!

Matt Farejowicz | Nice one

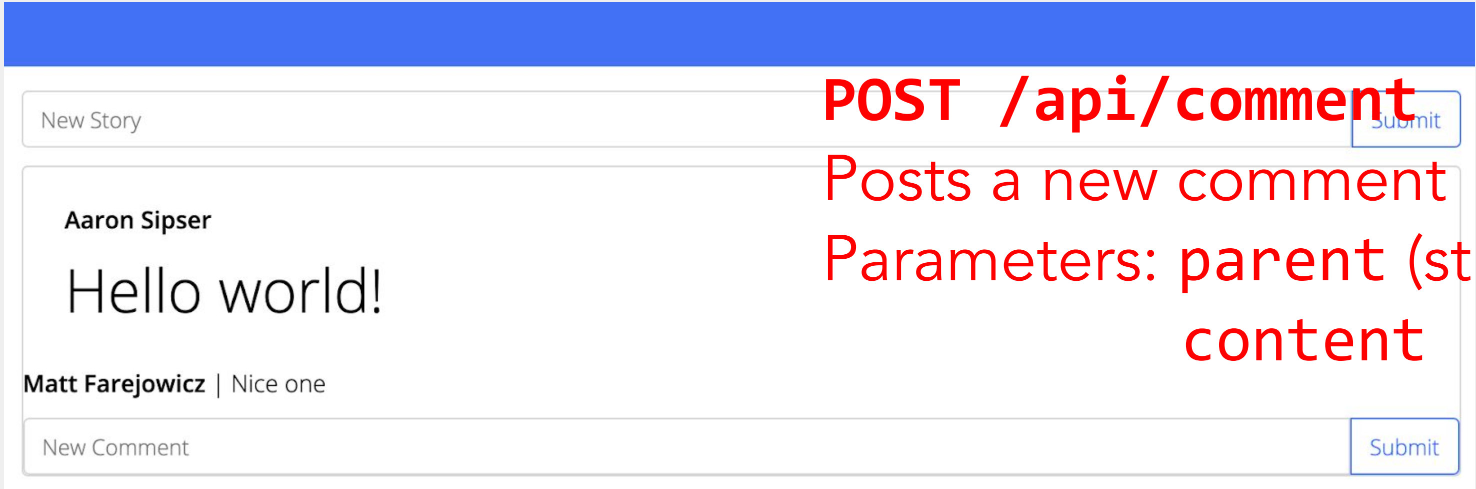
Your turn! Step 8

- Implement `NewComment` in `NewPostInput.js`
- Use `<NewComment (props) />` in `Card.js`

`GET /api/comment`

Returns all comments for a sto

Parameters: `parent` (string)



Aaron Sipser

Hello world!

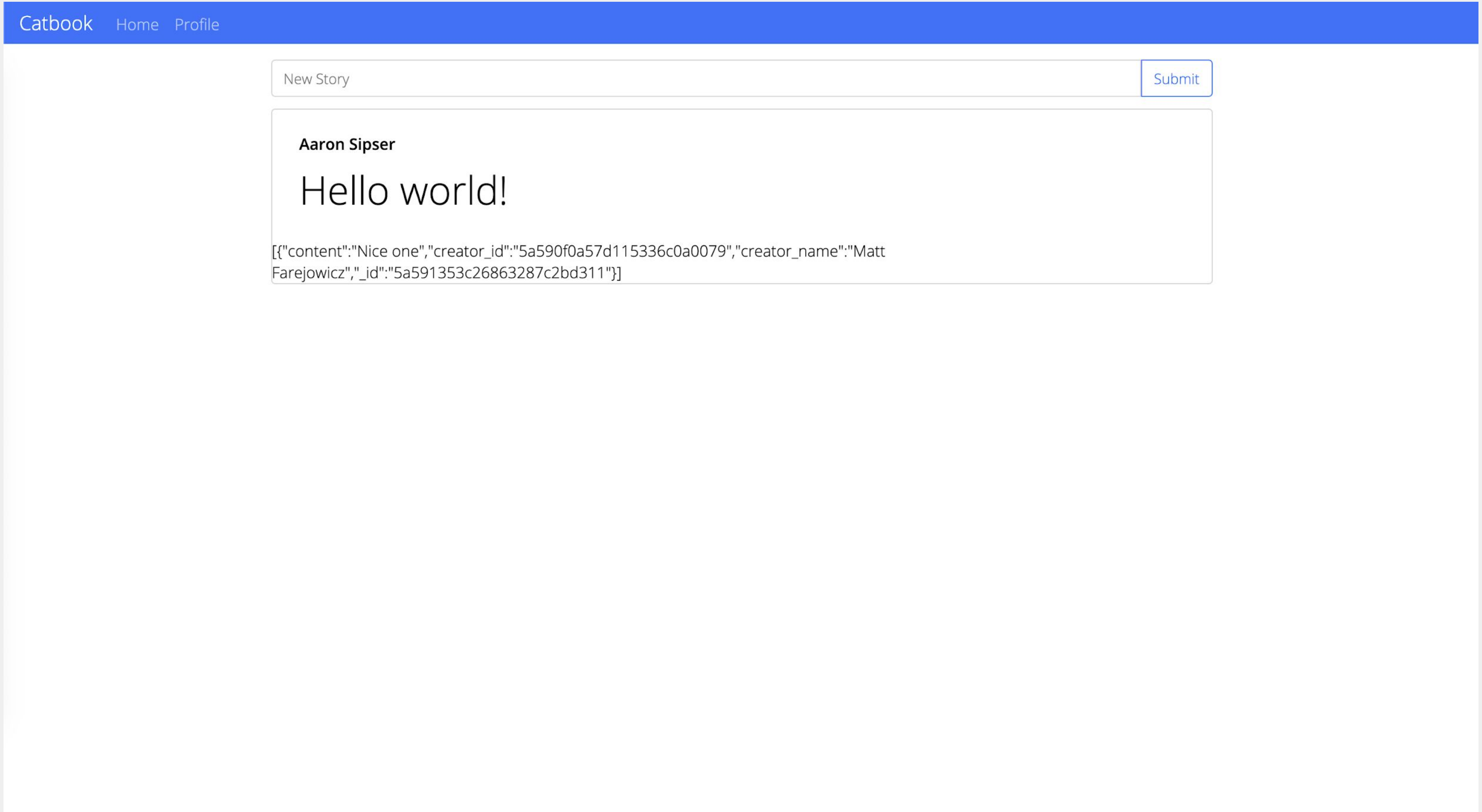
Matt Farejowicz | Nice one

`POST /api/comment`

Posts a new comment

Parameters: `parent` (string)
`content`

End of step 6

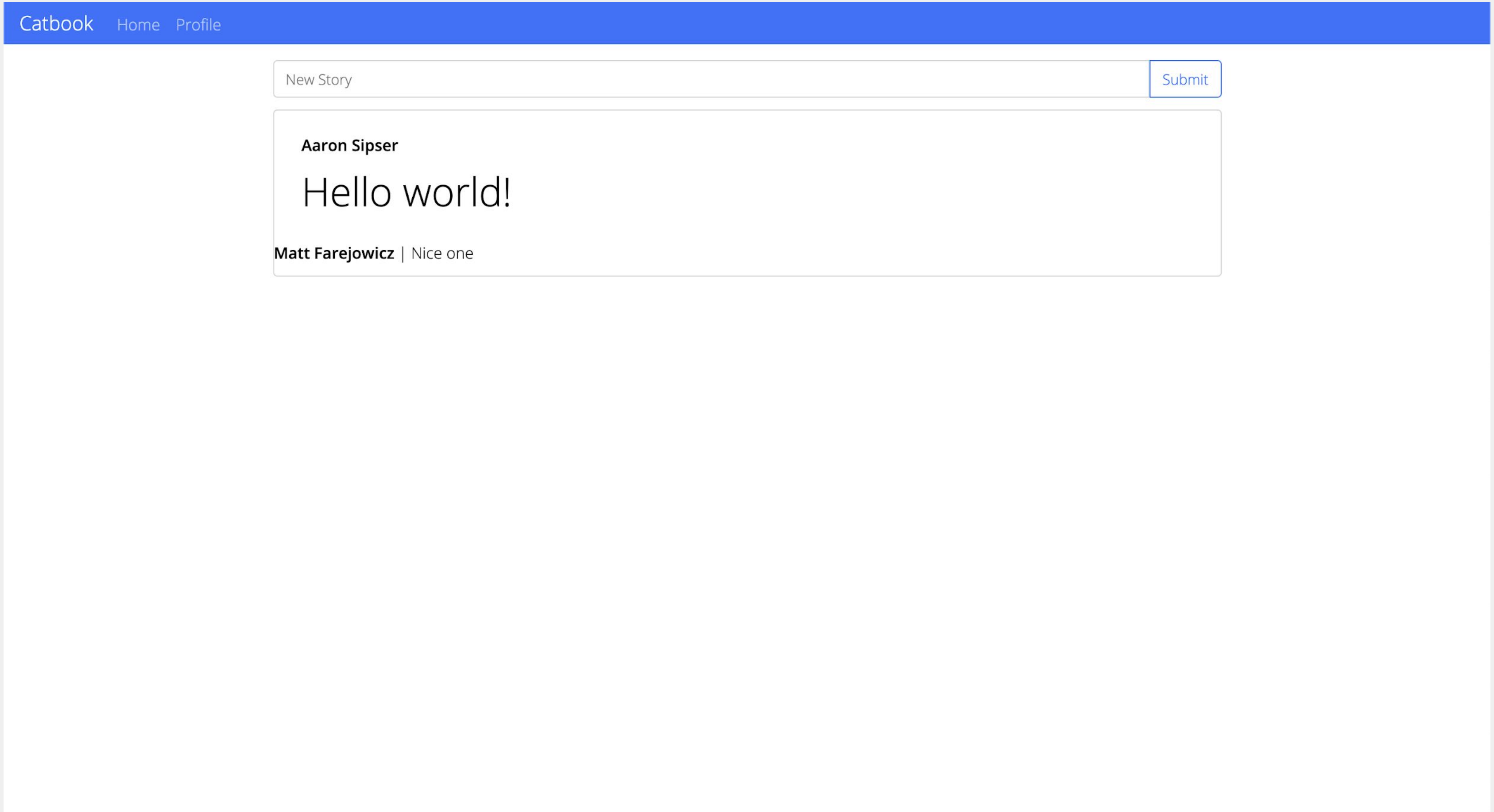


Aaron Sipser

Hello world!

[{"content":"Nice one","creator_id":"5a590f0a57d115336c0a0079","creator_name":"Matt
Farejowicz","_id":"5a591353c26863287c2bd311"}]

End of step 7



Catbook Home Profile

New Story

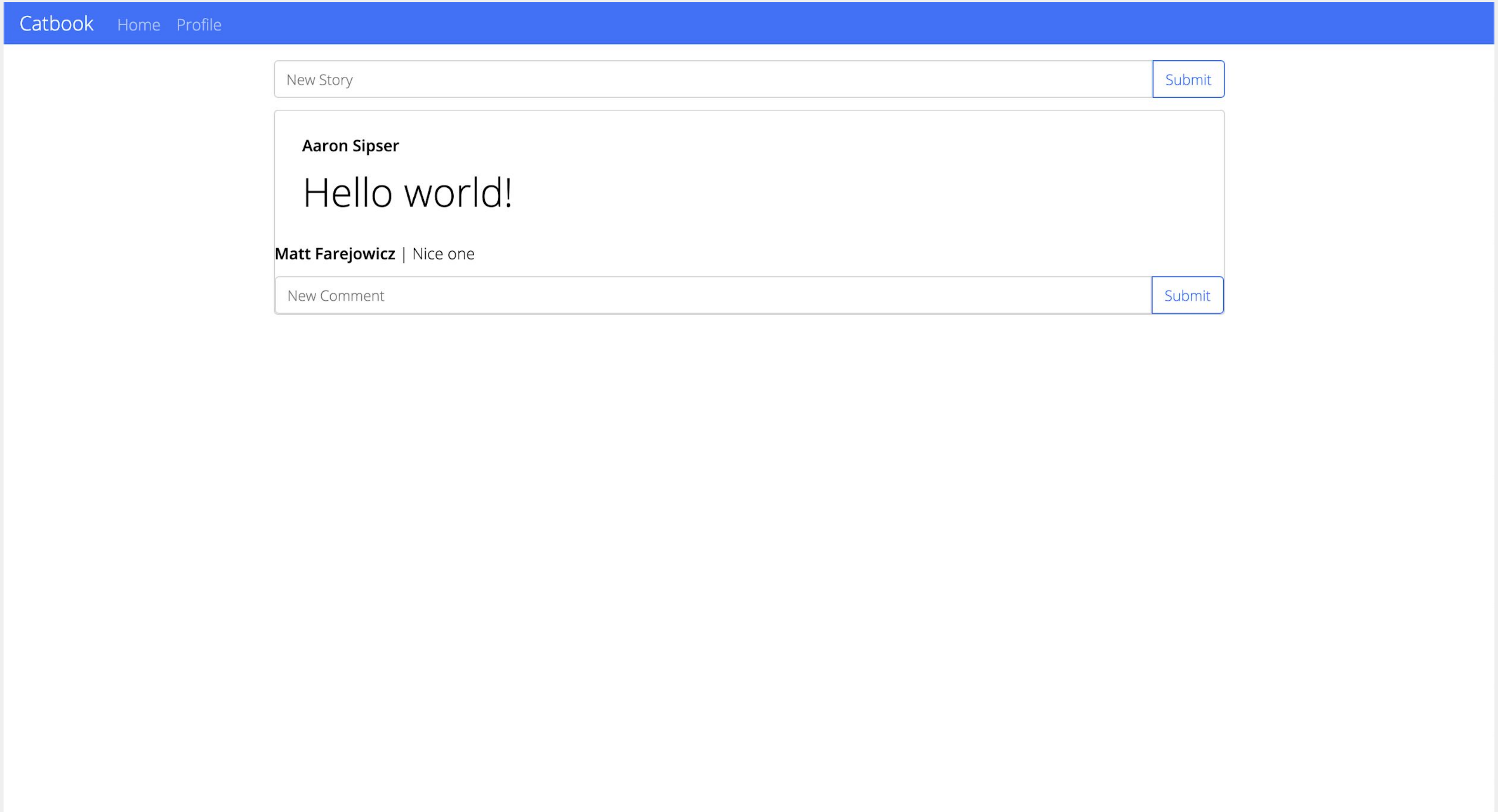
Submit

Aaron Sipser

Hello world!

Matt Farejowicz | Nice one

End of step 8



Aaron Sipser

Hello world!

Matt Farejowicz | Nice one

Your turn! Steps 6-8

- Your tasks:
  - Step 6:
    - GET comments in `Card.js`
    - Render them like stories
    - Use the `Card` in `Feed.js`
  - Step 7:
    - Implement `SingleComment.js`
    - Use `SingleComment` in `Card.js`
  - Step 8:
    - Implement `NewComment` in `NewPostInput.js`
    - Use `NewComment` in `Card.js`

- Tips:
  - Refer to `Feed.js`, `SingleStory.js`, and `NewStory.js`
  - Styles provided!
  - Ask for help!
  - `git reset --hard`
  - `git checkout w3-stepX`

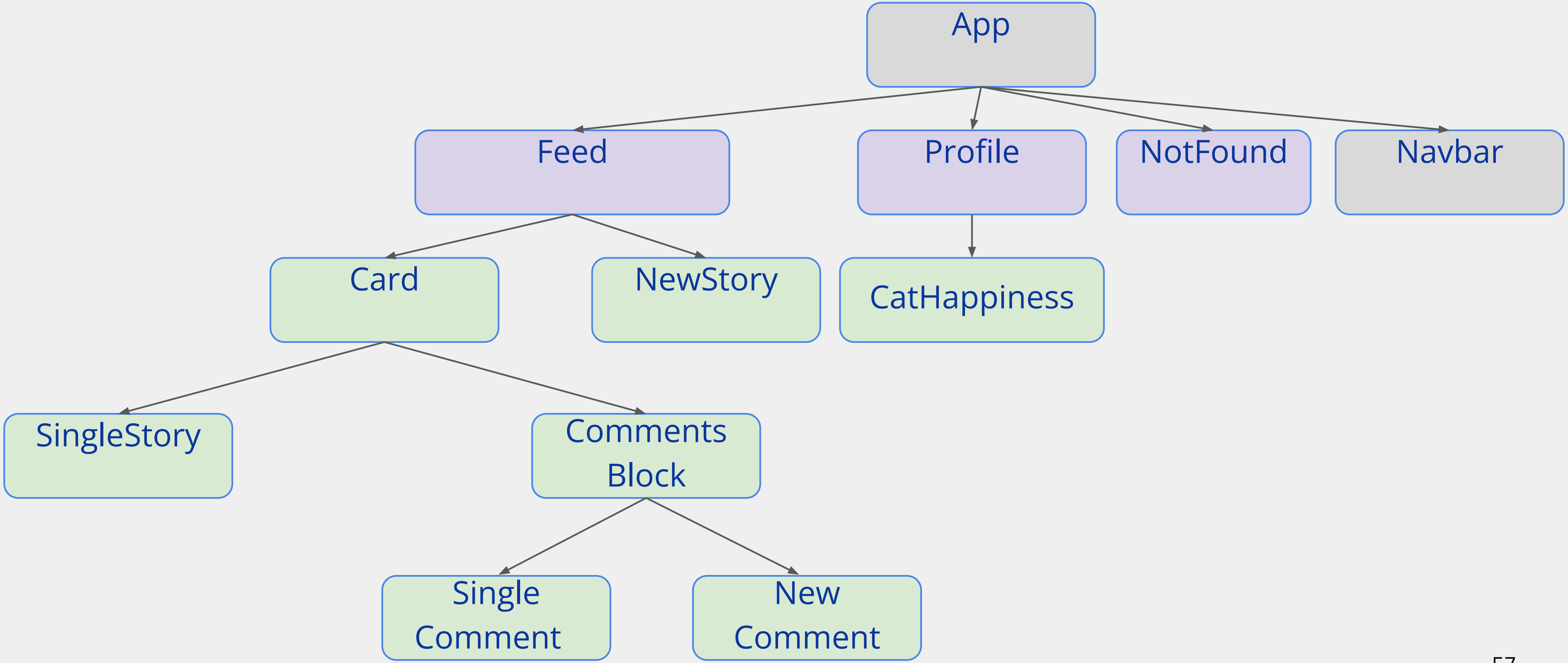
Let’s get on the same page

- Catch up:
  - `git reset --hard`
  - `git checkout w3-step9`

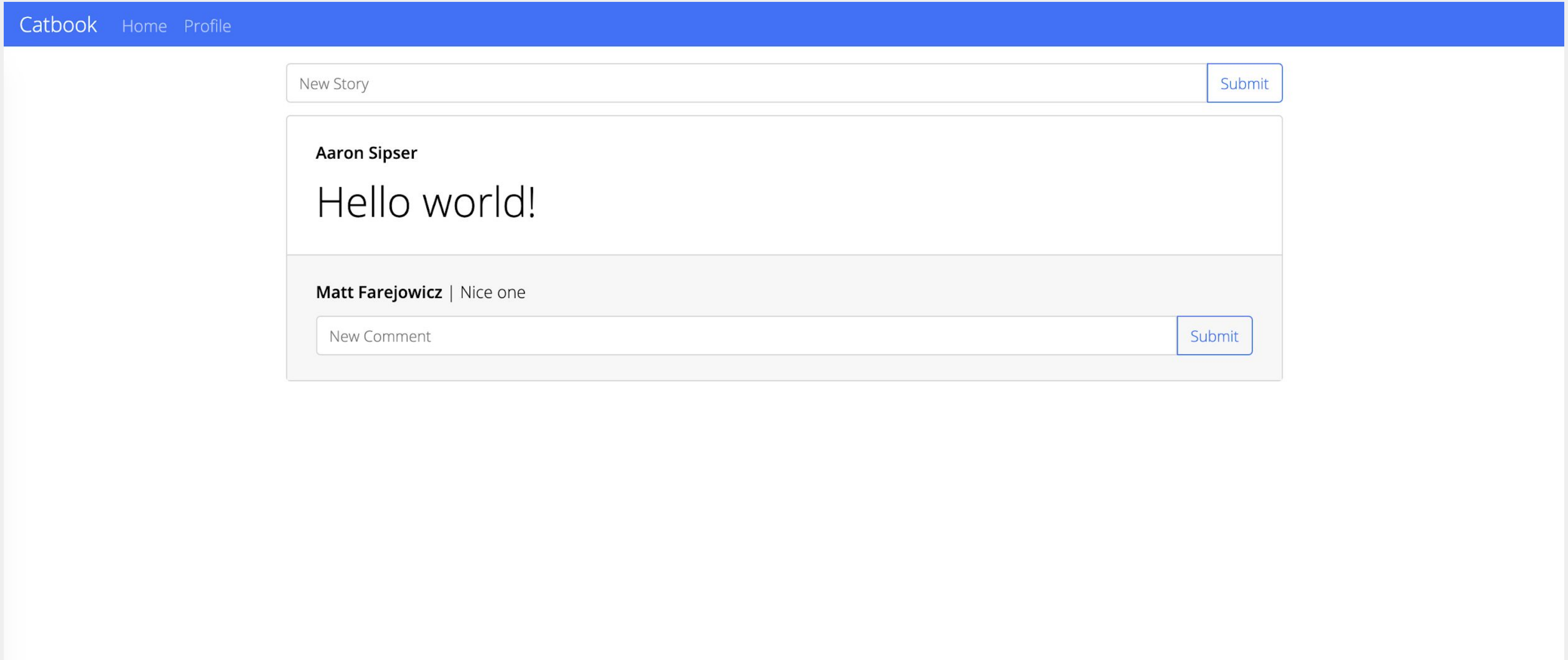
Follow along: step 9

- Extracting out components?
    - Want to keep components small and modular
- Let’s try and improve upon `Card.js`

Component hierarchy after today!



End of step 9



Aaron Sipser

Hello world!

Matt Farejowicz | Nice one

Homework: step 10

- Catch up:
  - `git reset --hard`
  - `git checkout w3-step10`

- What to do about needing to refresh?
- Let’s use the power of React!

Homework: step 10

- Do the same for comments!
- In `Card.js`
  - 1) Define an `addNewComment` function
  - 2) Pass as prop to `CommentsBlock`
- In `CommentsBlock.js`
  - 1) Pass `addNewComment` as prop to `NewComment`
- In `NewPostInput.js`
  - 1) Use `.then` to call `AddNewComment` after posting

The end!

- If you weren’t able to finish live updating, complete it at home.
- If you’d just like to see the solution:
    - `git reset --hard`
    - `git checkout w3-complete`