# how to code good*

Thanh Nguyen

*well (if you want to be grammatically correct)

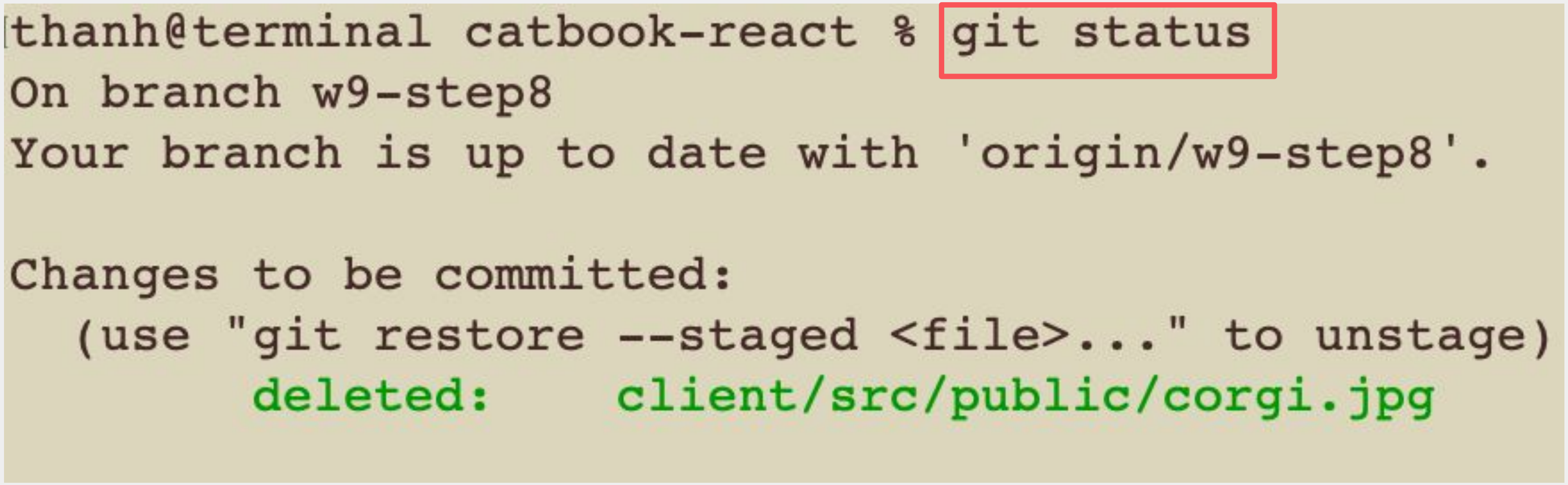
## Don’t use git like a monkey

## Before you start working

- `git pull`
  - Loads your teammates' changes
  - Do it frequently to avoid merge conflicts
- `git pull --rebase`
  - Puts your changes on top of the remote changes
  - I recommend doing this instead of just git pull…a bit more advanced
- `git pull --merge`
  - Default behavior
  - local changes are merged with the remote changes

## Before you commit

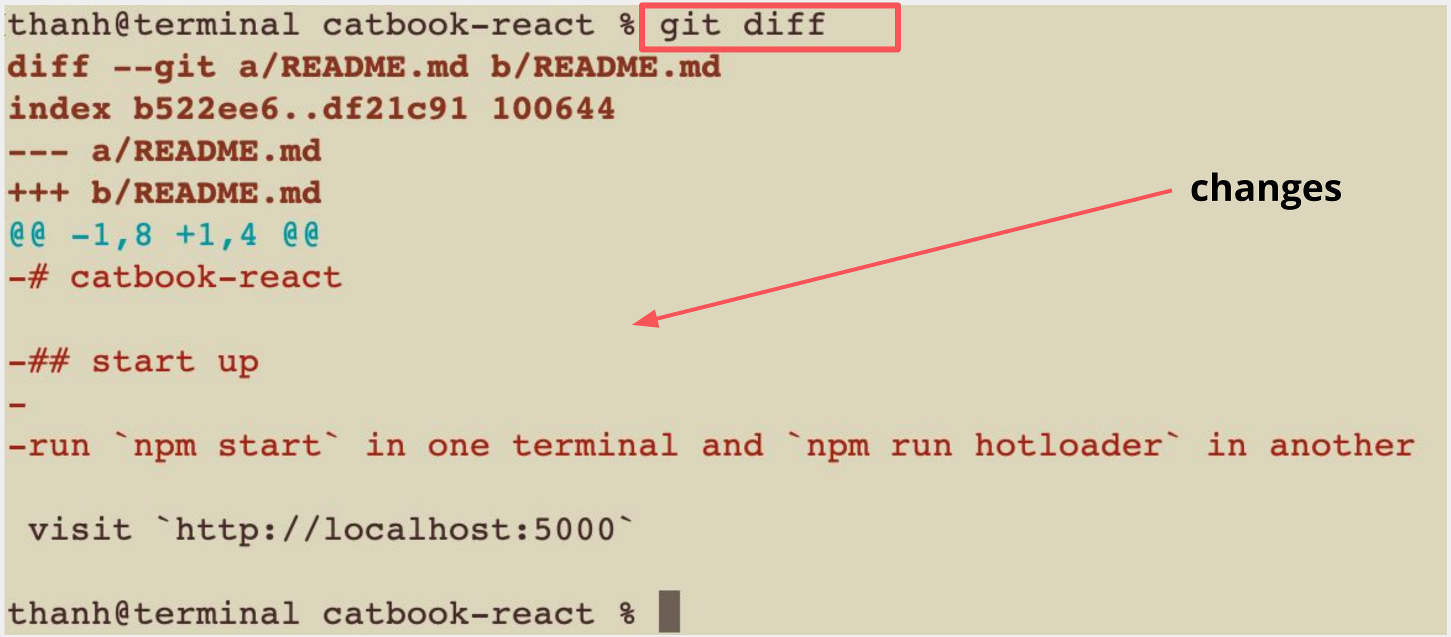
- Before running `git add` / `git commit`
- Review what changes you've made with `git status`



```
thanh@terminal catbook-react % git status
On branch w9-step8
Your branch is up to date with 'origin/w9-step8'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    client/src/public/corgi.jpg
```

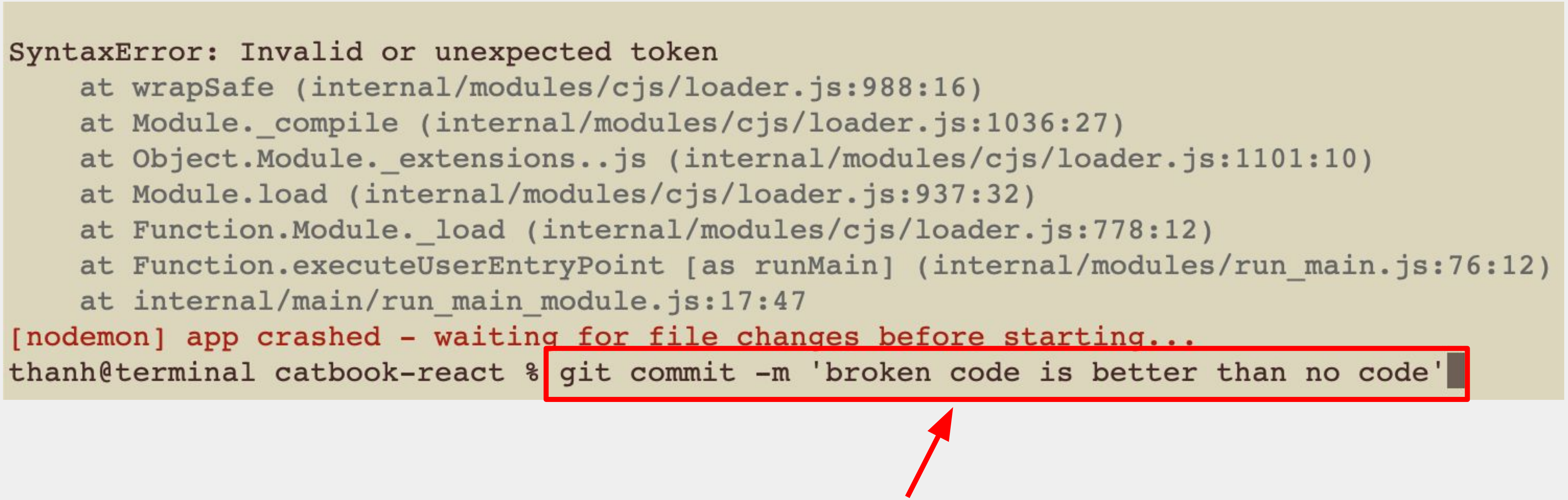
## Before you commit

- Optionally, review your changes more thoroughly with `git diff`



```
thanh@terminal catbook-react % git diff
diff --git a/README.md b/README.md
index b522ee6..df21c91 100644
--- a/README.md
+++ b/README.md
@@ -1,8 +1,4 @@
-# catbook-react

-## start up
-
-run `npm start` in one terminal and `npm run hotloader` in another

 visit `http://localhost:5000`

thanh@terminal catbook-react %
```

## Before you commit

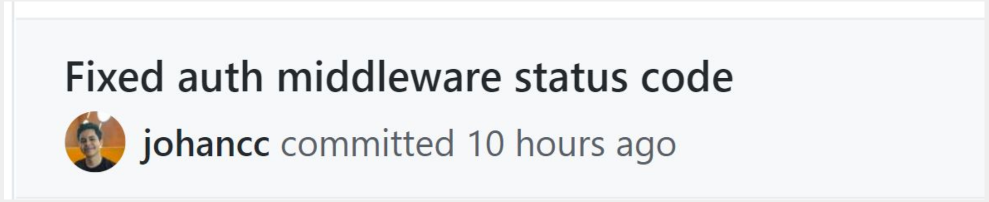


```
SyntaxError: Invalid or unexpected token
    at wrapSafe (internal/modules/cjs/loader.js:988:16)
    at Module._compile (internal/modules/cjs/loader.js:1036:27)
    at Object.Module._extensions..js (internal/modules/cjs/loader.js:1101:10)
    at Module.load (internal/modules/cjs/loader.js:937:32)
    at Function.Module._load (internal/modules/cjs/loader.js:778:12)
    at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:76:12)
    at internal/main/run_main_module.js:17:47
[nodemon] app crashed - waiting for file changes before starting...
thanh@terminal catbook-react % git commit -m 'broken code is better than no code'
```

**DO NOT!**

# When to commit

- When you add something and it works, then commit
  - More often is generally better than less often

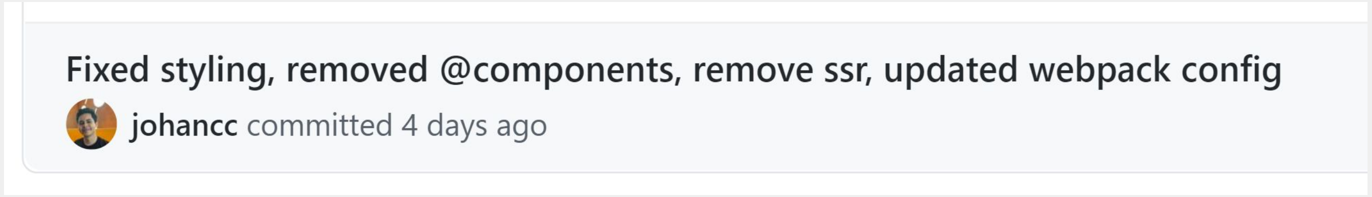


Fixed auth middleware status code

johancc committed 10 hours ago

Commit adds a single thing, and is focused/clear

- Avoid mega-commits



Fixed styling, removed @components, remove ssr, updated webpack config

johancc committed 4 days ago

Commit tries to do too much, not focused

## DANGEROUS commands!

- `git add .`
  - This will add all files in the current directory to git
  - Make sure to setup .gitignore files to avoid committing node_modules or credentials
- `git push --force`
  - **NEVER USE** unless you know what you're doing
- `git reset --hard`
  - Deletes all uncommitted changes
- `git commit --amend`
  - Don't ammend commits you've already pushed
  - Make a new commit instead

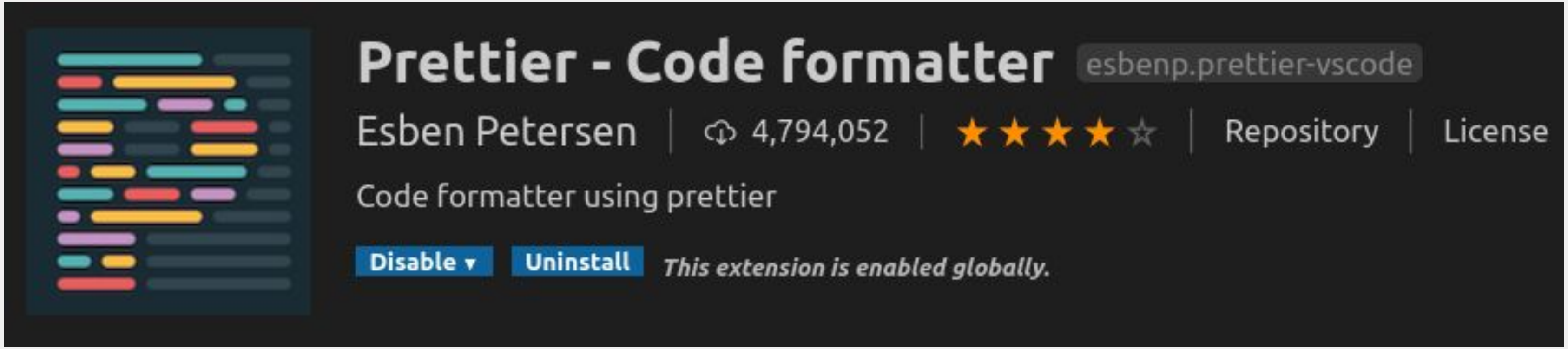
## Useful git commands

- `git stash`
  - git stash allows you to store your current changes
- `git stash pop`
  - git stash allows you to “pop” the most recent stored changes
- `git branch`
  - git branch allows you to “diverge” and focus on one issue
  - git checkout (used in workshops!) will then allow you to move to different branches

## Code formatting/style

## Use prettier!!

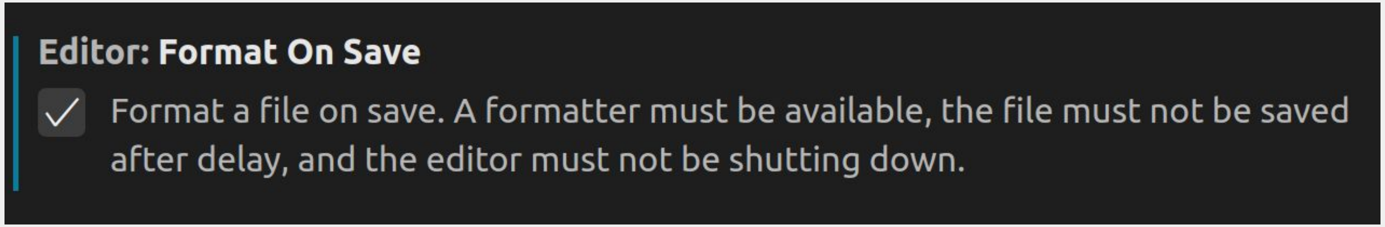
- VSCode extension


Prettier - Code formatter esbenp.prettier-vscode

Esben Petersen | 4,794,052 | ★★★★☆ | Repository | License

Code formatter using prettier

*This extension is enabled globally.*

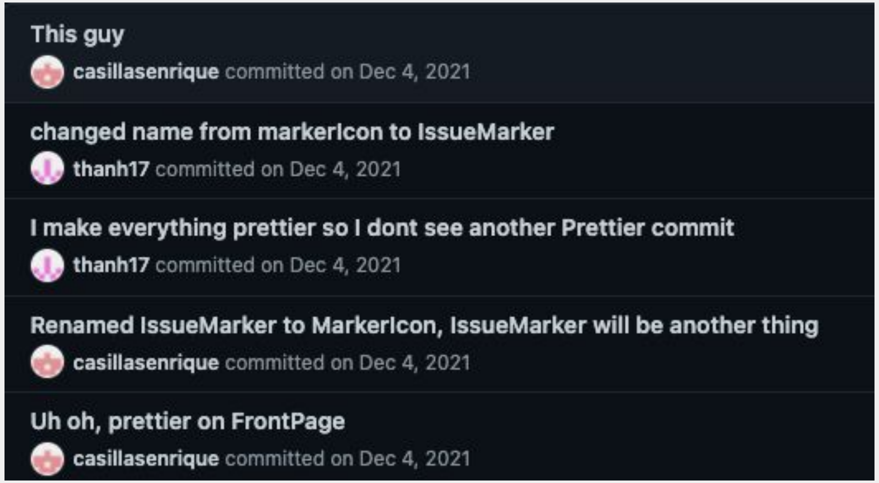

**Editor: Format On Save**

✓ Format a file on save. A formatter must be available, the file must not be saved after delay, and the editor must not be shutting down.

- Makes your code pretty every time you hit “save”

## Danger!

- Decide as a team if you want to use prettier
- Either everybody use it, or nobody use it
- **Can run into nasty merge conflicts otherwise**


This guy
casillasenrique committed on Dec 4, 2021

changed name from markerIcon to IssueMarker
thanh17 committed on Dec 4, 2021

I make everything prettier so I dont see another Prettier commit
thanh17 committed on Dec 4, 2021

Renamed IssueMarker to MarkerIcon, IssueMarker will be another thing
casillasenrique committed on Dec 4, 2021

Uh oh, prettier on FrontPage
casillasenrique committed on Dec 4, 2021

Merge Conflict

Monkey moment from Thanh
(inconsistent Prettier package)

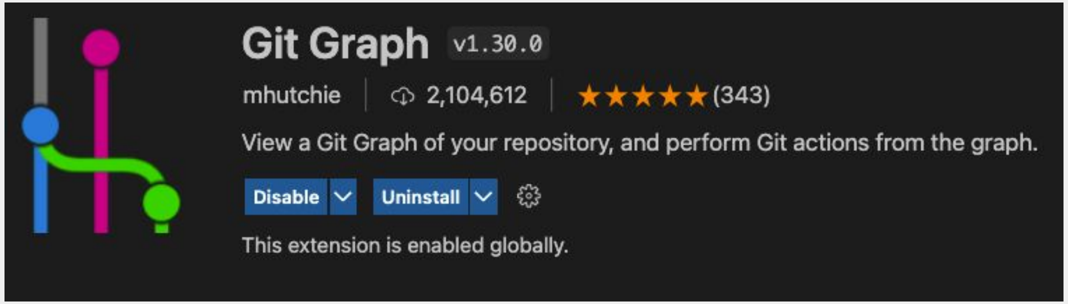
Autoformat with Prettier

## Some Support!

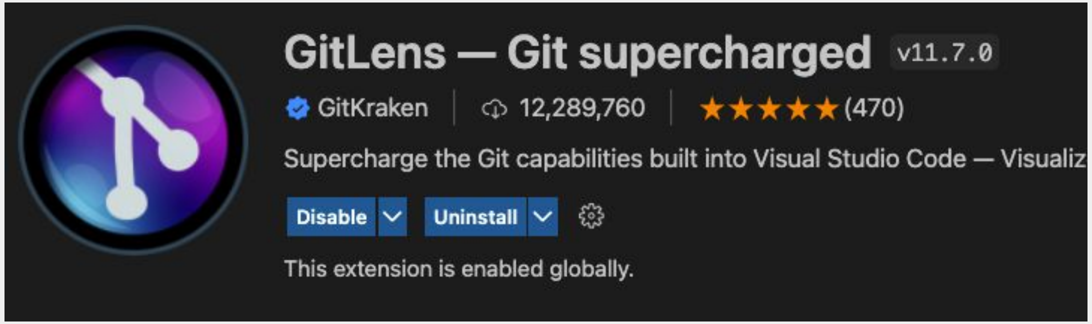
*but still make sure you know how to

## Git Support

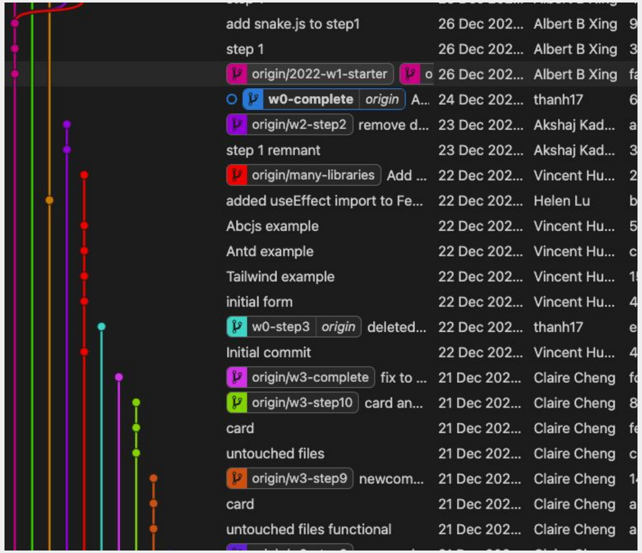
- Don’t like terminal? Don’t use it!*


Git Graph v1.30.0
mhutchie | 2,104,612 | ★★★★★ (343)
View a Git Graph of your repository, and perform Git actions from the graph.

This extension is enabled globally.


GitLens — Git supercharged v11.7.0
GitKraken | 12,289,760 | ★★★★★ (470)
Supercharge the Git capabilities built into Visual Studio Code — Visualiz

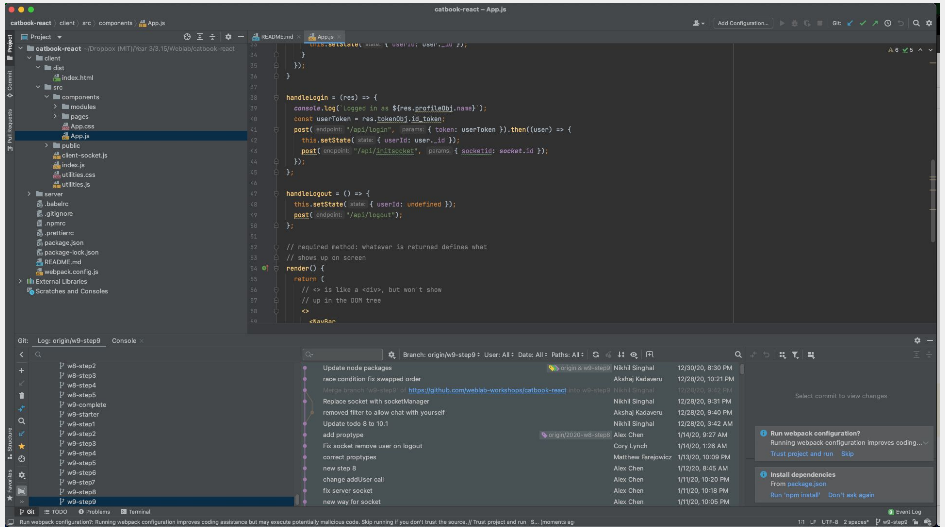
This extension is enabled globally.



| | | | |
|---|---|---|---|
| add snake.js to step1 | 26 Dec 202... | Albert B Xing | 9 |
| step 1 | 26 Dec 202... | Albert B Xing | 3 |
| origin/2022-w1-starter o | 26 Dec 202... | Albert B Xing | fa |
| w0-complete origin A... | 24 Dec 202... | thanh17 | 6 |
| origin/w2-step2 remove d... | 23 Dec 202... | Akshaj Kad... | a |
| step 1 remnant | 23 Dec 202... | Akshaj Kad... | 3 |
| origin/many-libraries Add ... | 22 Dec 202... | Vincent Hu... | 2 |
| added useEffect import to Fe... | 22 Dec 202... | Helen Lu | b |
| Abcjs example | 22 Dec 202... | Vincent Hu... | 5 |
| Antd example | 22 Dec 202... | Vincent Hu... | c |
| Tailwind example | 22 Dec 202... | Vincent Hu... | 1 |
| initial form | 22 Dec 202... | Vincent Hu... | 4 |
| w0-step3 origin deleted... | 22 Dec 202... | thanh17 | e |
| Initial commit | 22 Dec 202... | Vincent Hu... | 4 |
| origin/w3-complete fix to ... | 21 Dec 202... | Claire Cheng | fo |
| origin/w3-step10 card an... | 21 Dec 202... | Claire Cheng | 8 |
| card | 21 Dec 202... | Claire Cheng | fe |
| untouched files | 21 Dec 202... | Claire Cheng | c |
| origin/w3-step9 newcom... | 21 Dec 202... | Claire Cheng | 1 |
| card | 21 Dec 202... | Claire Cheng | a |
| untouched files functional | 21 Dec 202... | Claire Cheng | a |

# A Different Editor..?

- Webstorm/IntelliJ
  - A lot more resource intensive…RAM goes brrrrrrrrr
  - A lot more features…so can be confusing
  - Better code and specification assistance…and superior to VSCode (In my opinion)
  - Talk to me if you want to know more…



catbook-react – App.js

```
      }
    });
  }

  handleLogin = (res) => {
    console.log(`Logged in as ${res.profileObj.name}`);
    const userToken = res.tokenObj.id_token;
    post( endpoint: "/api/login",  params: { token: userToken }).then((user) => {
      this.setState( state: { userId: user._id });
      post( endpoint: "/api/initsocket",  params: { socketid: socket.id });
    });
  };

  handleLogout = () => {
    this.setState( state: { userId: undefined });
    post( endpoint: "/api/logout");
  };

  // required method: whatever is returned defines what
  // shows up on screen
  render() {
    return (
      // <> is like a <div>, but won't show
      // up in the DOM tree
      <>
        <NavBar
```

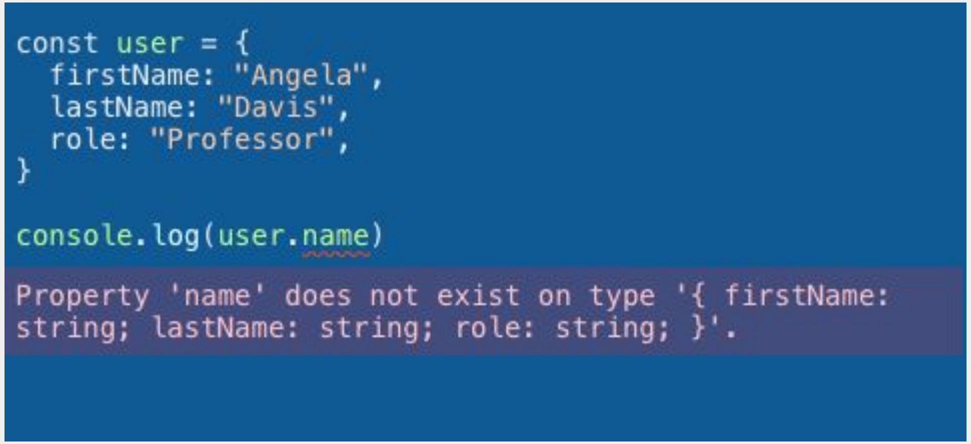
```
Branch: origin/w9-step9  User: All  Date: All  Paths: All
Update node packages                      origin & w9-step9   Nikhil Singhal       12/30/20, 8:30 PM
race condition fix swapped order                              Akshaj Kadaveru      12/28/20, 10:21 PM
Merge branch 'w9-step9' of https://github.com/weblab-workshops/catbook-react into w9-step9   Nikhil Singhal   12/28/20, 9:42 PM
Replace socket with socketManager                             Nikhil Singhal       12/28/20, 9:31 PM
removed filter to allow chat with yourself                    Akshaj Kadaveru      12/28/20, 9:40 PM
Update todo 8 to 10.1                                         Nikhil Singhal       12/28/20, 3:42 AM
add proptype                              origin/2020-w8-step8 Alex Chen           1/14/20, 9:27 AM
Fix socket remove user on logout                              Cory Lynch           1/14/20, 1:26 AM
correct proptypes                                             Matthew Farejowicz   1/13/20, 10:09 PM
new step 8                                                    Alex Chen            1/12/20, 8:45 AM
change addUser call                                           Alex Chen            1/11/20, 10:20 PM
fix server socket                                             Alex Chen            1/11/20, 10:18 PM
new way for socket                                            Alex Chen            1/11/20, 10:05 PM
```

Run webpack configuration? Running webpack configuration improves coding… Trust project and run  Skip

Install dependencies From package.json Run 'npm install'  Don't ask again

Run webpack configuration?: Running webpack configuration improves coding assistance but may execute potentially malicious code. Skip running if you don't trust the source. // Trust project and run  S... (moments ago)

## Typescript!! (Tomorrow with Vincent)



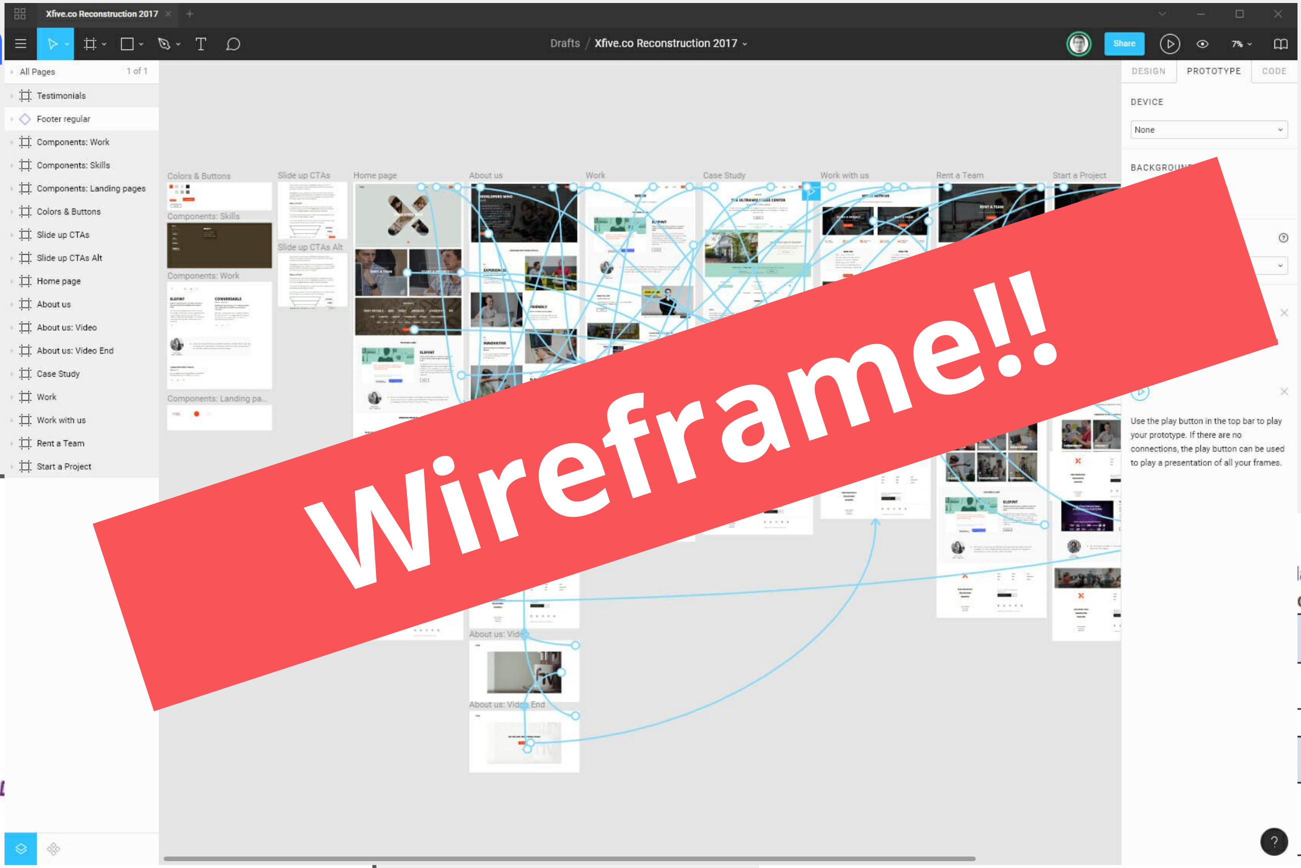
```
const user = {
  firstName: "Angela",
  lastName: "Davis",
  role: "Professor",
}

console.log(user.name)
```

```
Property 'name' does not exist on type '{ firstName:
string; lastName: string; role: string; }'.
```

## Writing quality code

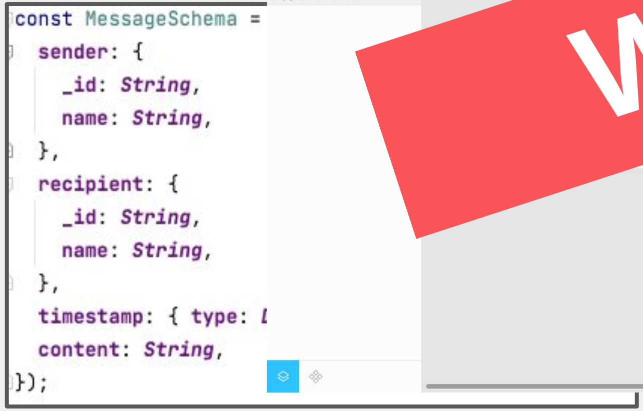
In most cases, bad code is a result of bad design.

# Design



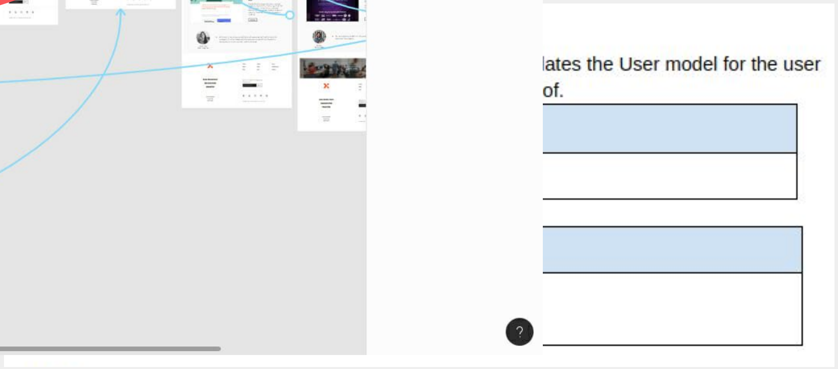
Use the play button in the top bar to play your prototype. If there are no connections, the play button can be used to play a presentation of all your frames.

**Wireframe!!**



```
const MessageSchema =
  sender: {
    _id: String,
    name: String,
  },
  recipient: {
    _id: String,
    name: String,
  },
  timestamp: { type: I
  content: String,
});
```

Mongoose schemas



lates the User model for the user
of.

API Specifications

“if you don't have any design, *you don't know what you are making.*”

- some dude from stackoverflow

## Scenario: I want to add a new feature, but...

My code is structured in a way that makes this difficult.

Two options:

1. Restructure the code, which might take **3 hours**
2. Come up with a hacky solution, which only takes **30 mins**

## What is hacky code?

Code may be hacky if…

- It’s super inefficient
- Difficult to understand/complicated logic flow
- Might easily break if you need to tweak it in the future
- Messes with a library in a way that wasn’t intended
- and so on…

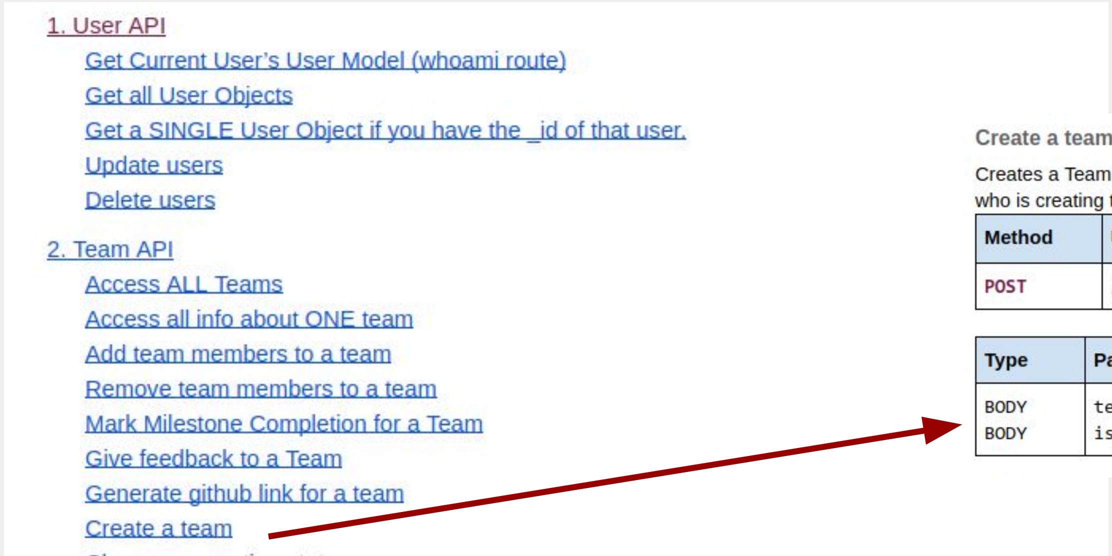
## Avoid the need for hacks at all

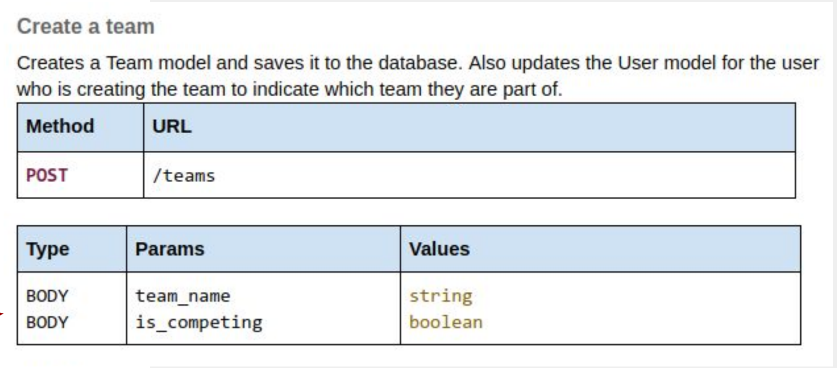
- **Design well from the start**
    - Keep in mind your designs will likely change!
    - Feel free to talk to us about this

- Draw out a React component hierarchy
- Discuss the best Mongoose schema with your team
- Write a **specification** for your API

# API Specification

- For every API route
  - Is it GET or POST (or something else?)
  - What parameters does it expect
  - What does it return?





Create a team

Creates a Team model and saves it to the database. Also updates the User model for the user who is creating the team to indicate which team they are part of.

| Method | URL |
|---|---|
| `POST` | `/teams` |

| Type | Params | Values |
|---|---|---|
| BODY | `team_name` | `string` |
| BODY | `is_competing` | `boolean` |

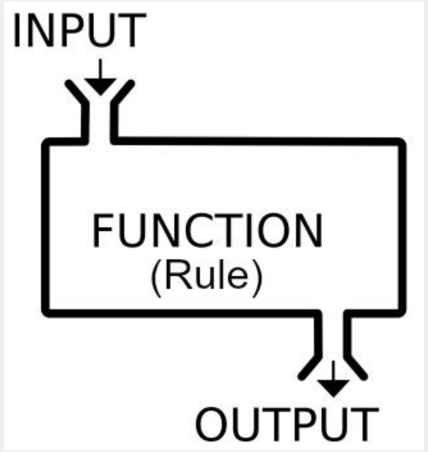
## What is clean code?

## D.R.Y. Principle

- Don’t repeat yourself!
- If you are repeating code anywhere, you probably want to abstract it into a function

## Functions

- Function should only do one thing: they should do it the right way and just do it.
  - A boolean parameter already clearly states that it does more than one thing.
- Clearly define your **specifications**
  - **Preconditions**? What inputs does it take?
  - **Postconditions**? What does your function return?
- Keep it simple stupid



## Meaningful Names

- Correctly describes what it does. The name does not imply “side effects”

## Signs/consequences of janky code

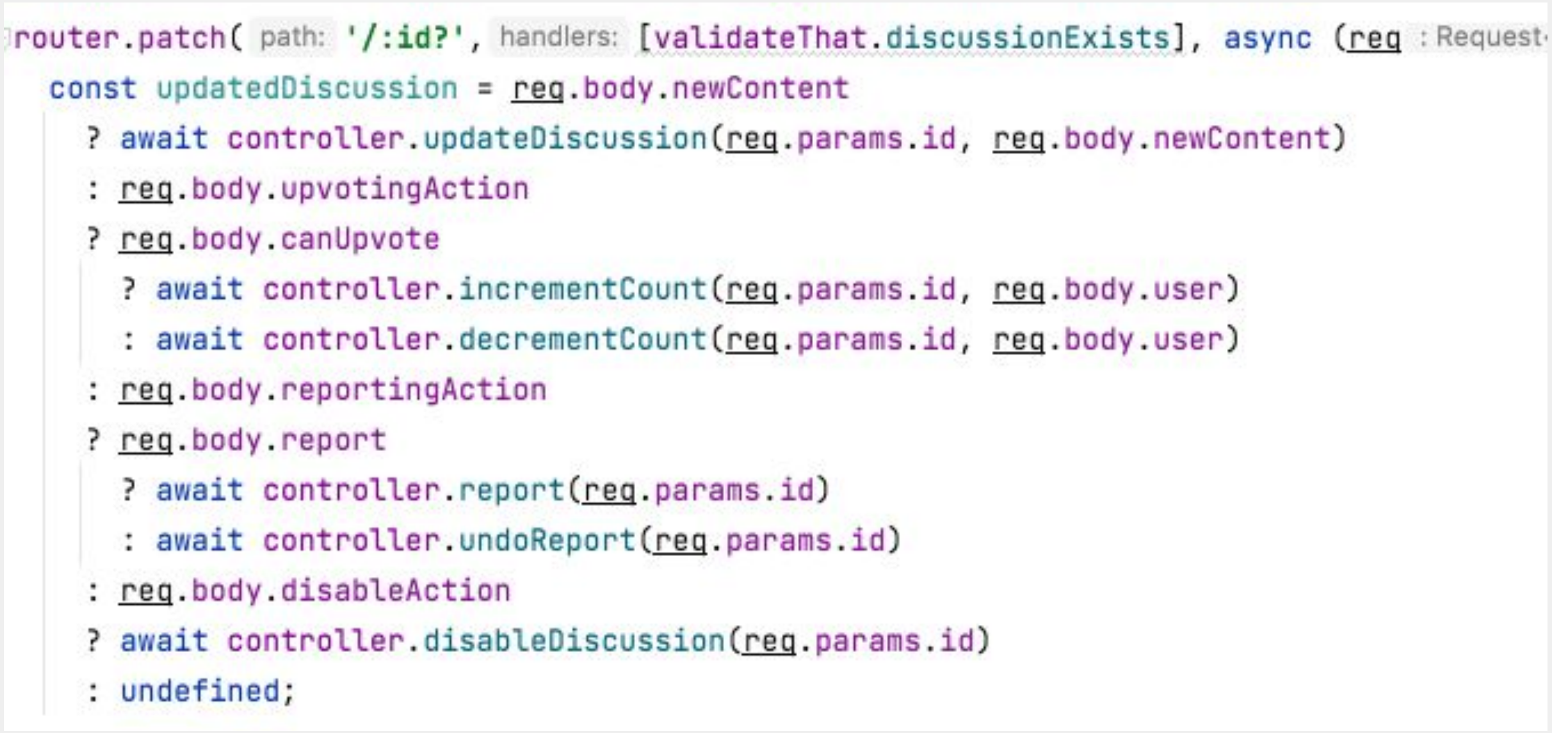
## Code Smell

Signs that your code may have some deeper problems...

- Lots of **copy-pasted** code
- Functions that are **way too long**
- Lines of code that are **way too long**
- **Dead code**: Entire functions/files that are unused

http://weblab.to/smell

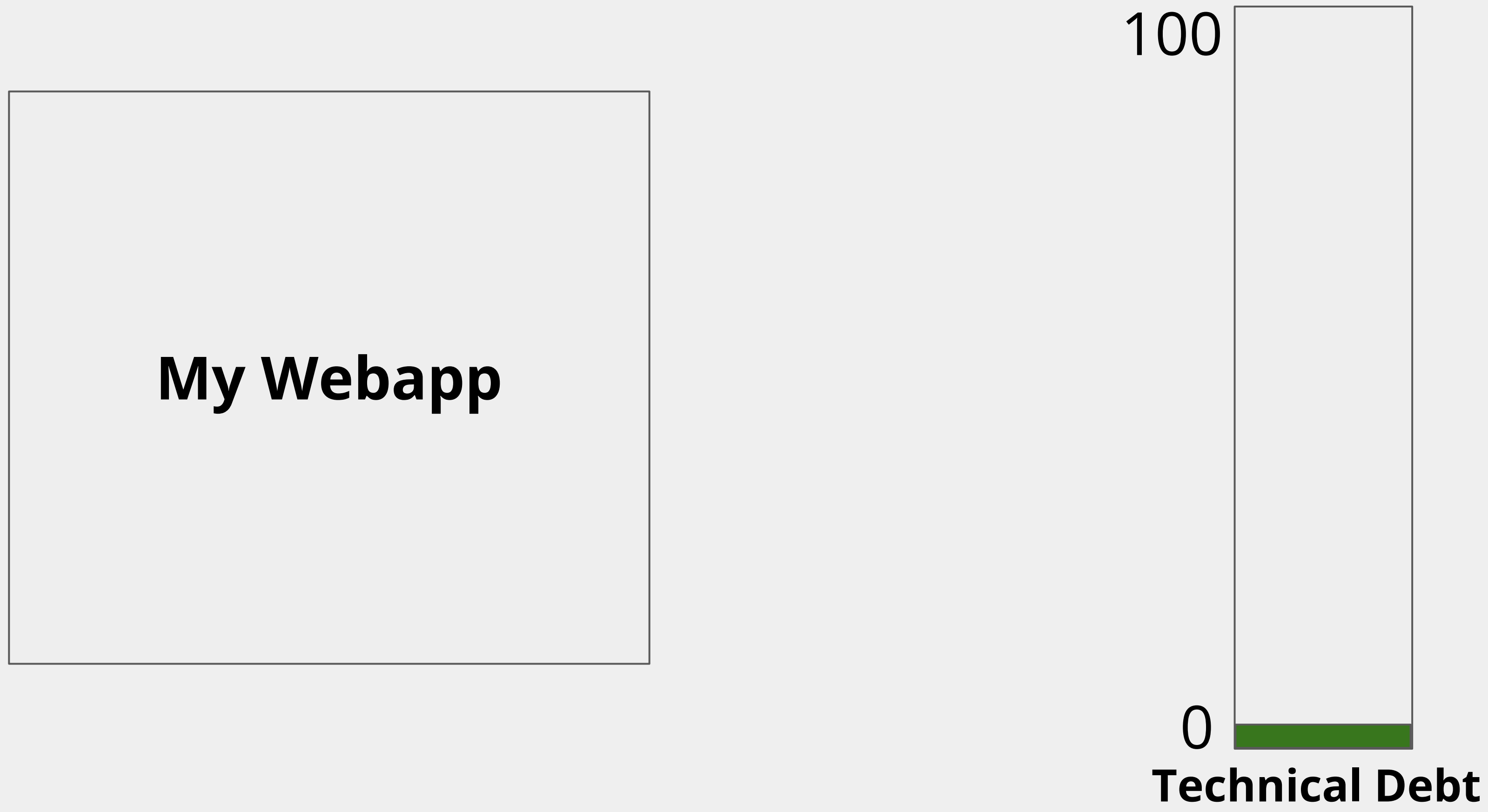
## Example: My past code



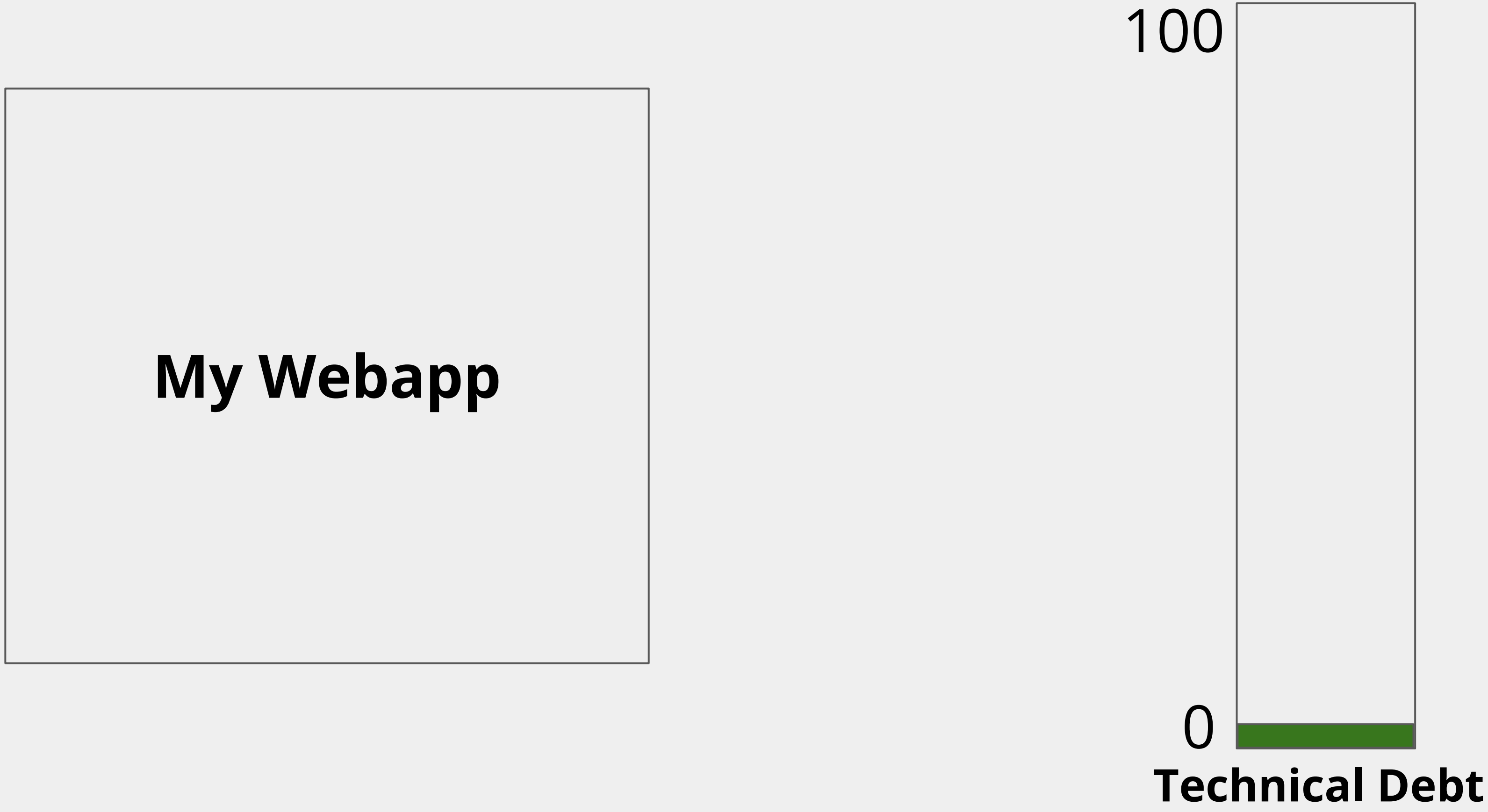
```
router.patch( path: '/:id?', handlers: [validateThat.discussionExists], async (req : Request
  const updatedDiscussion = req.body.newContent
    ? await controller.updateDiscussion(req.params.id, req.body.newContent)
    : req.body.upvotingAction
    ? req.body.canUpvote
      ? await controller.incrementCount(req.params.id, req.body.user)
      : await controller.decrementCount(req.params.id, req.body.user)
    : req.body.reportingAction
    ? req.body.report
      ? await controller.report(req.params.id)
      : await controller.undoReport(req.params.id)
    : req.body.disableAction
    ? await controller.disableDiscussion(req.params.id)
    : undefined;
```

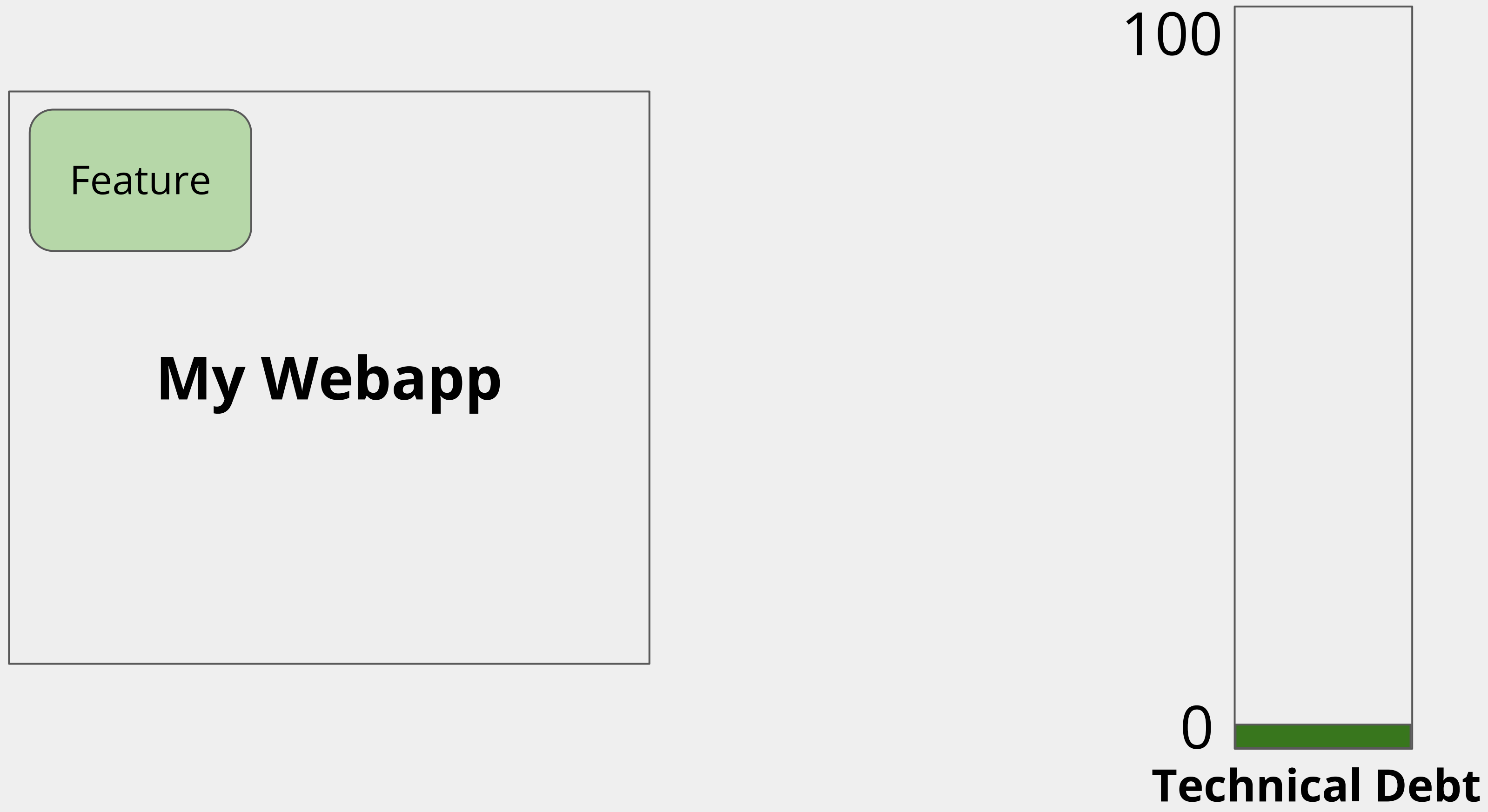
## Technical Debt

- When you choose a **quick hack** instead of a better solution
- Causes:
  - Lack of time
  - Laziness
  - Developer doesn’t know any better
- Creates “debt” you may need to pay off later…

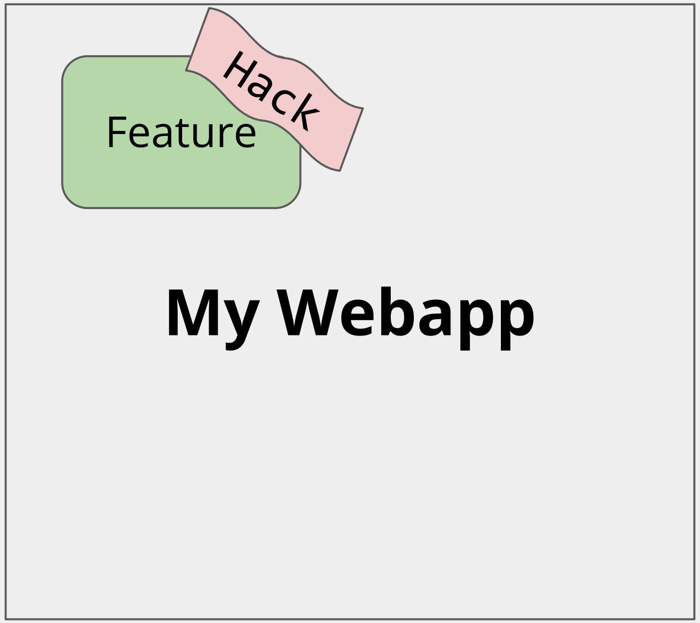

My Webapp
100
0
Technical Debt

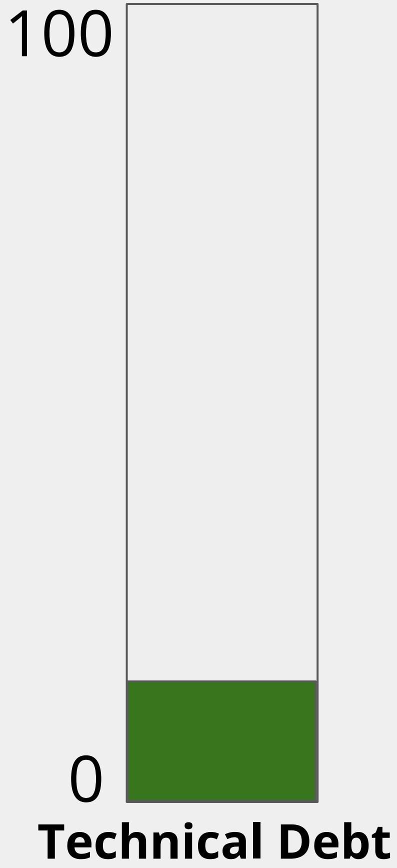
I’m gonna make feature X



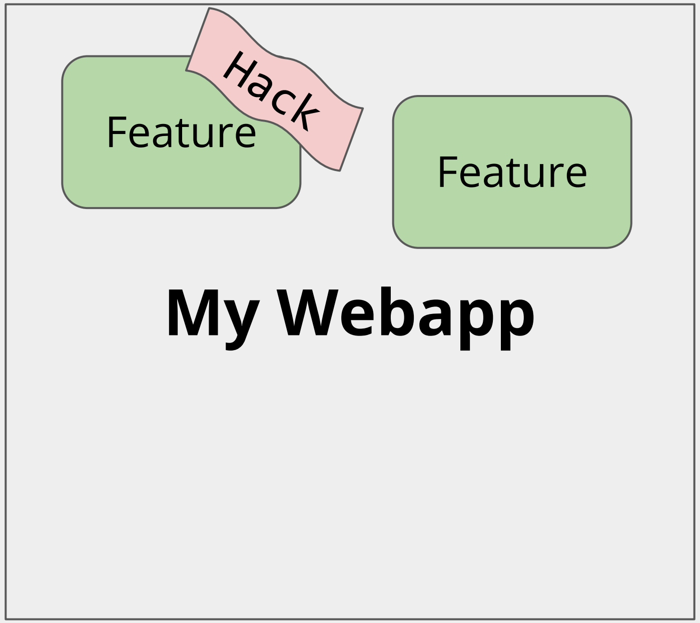

Feature
My Webapp
100
0
Technical Debt

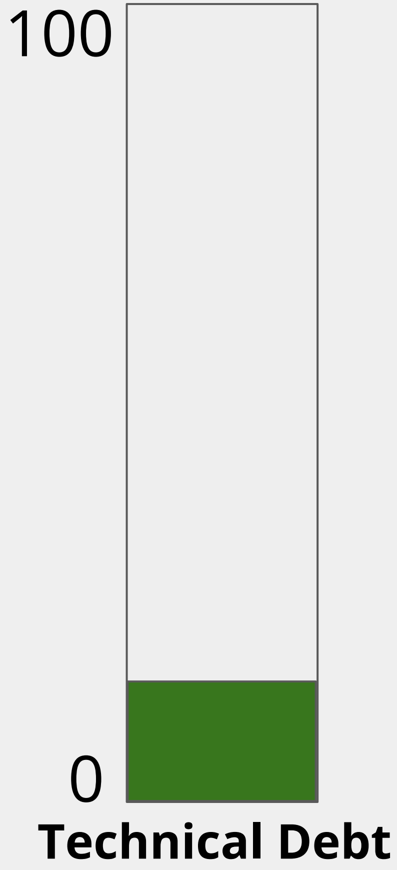
## It’s not quite right… let me add a hacky fix



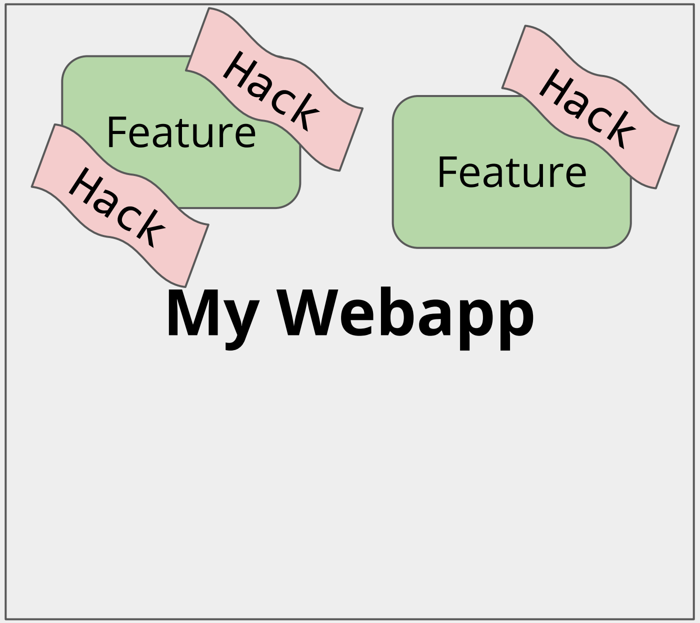


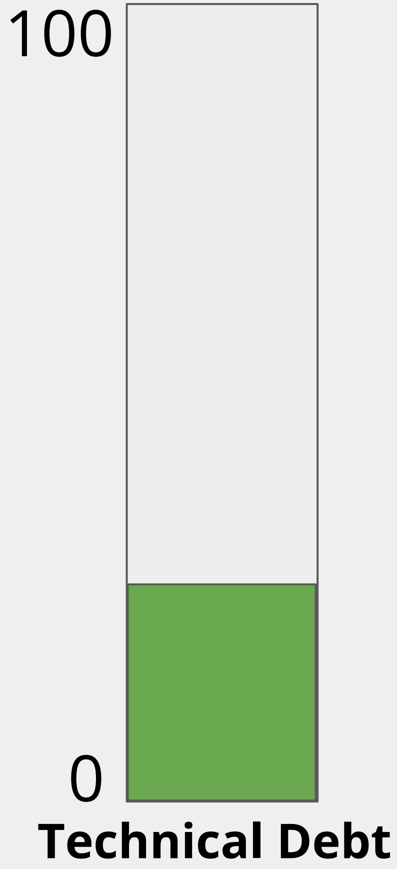
## Let’s add some more features

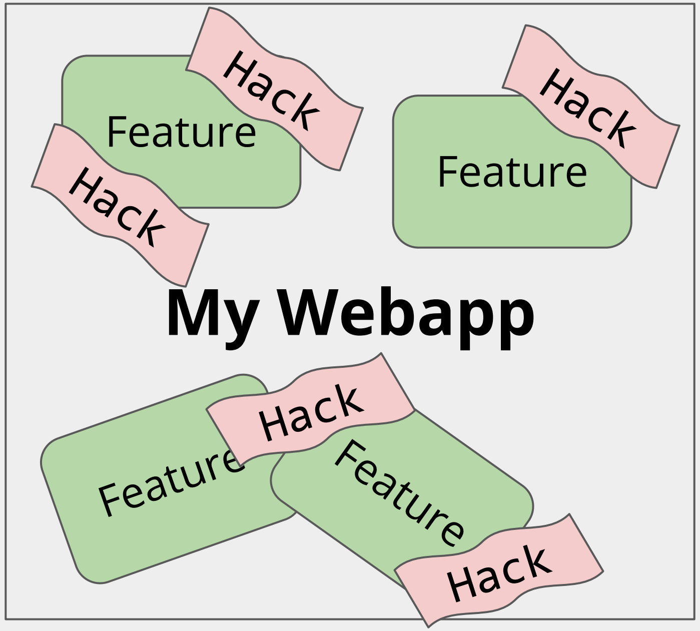


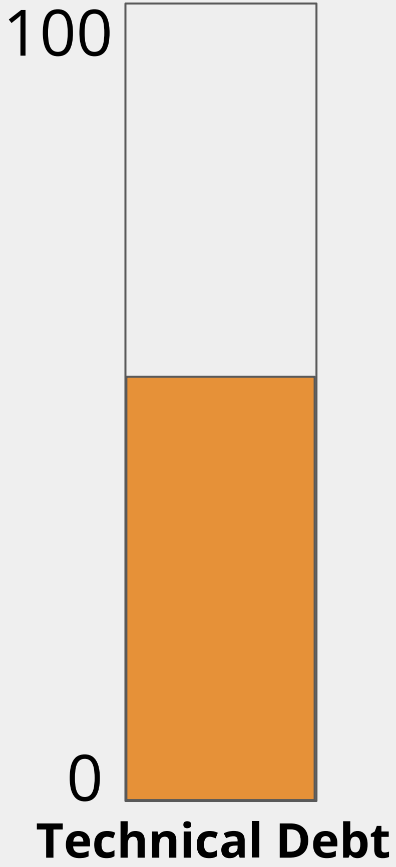


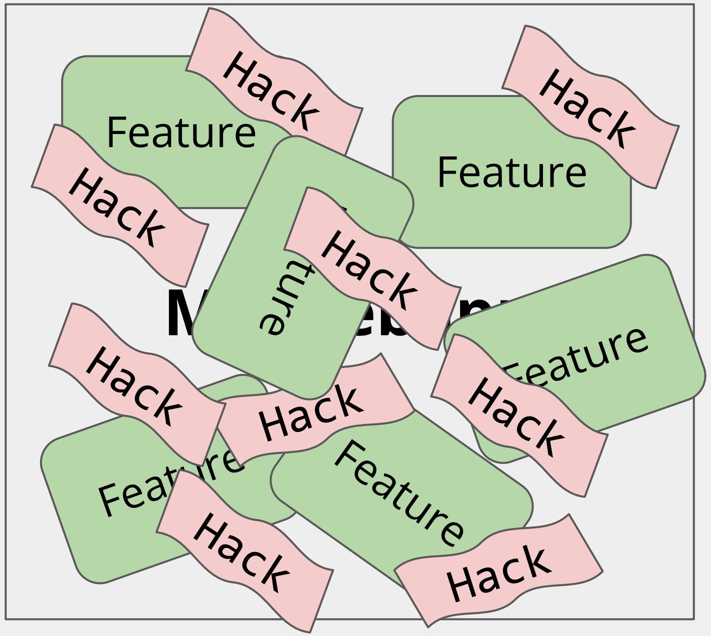
## I need some more hacks...

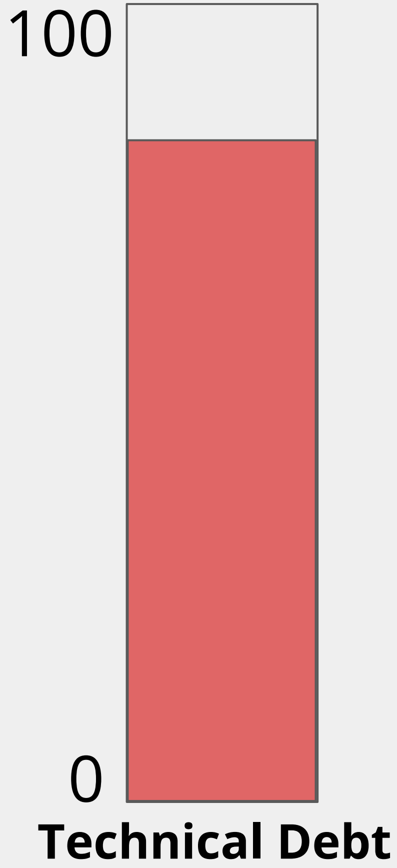




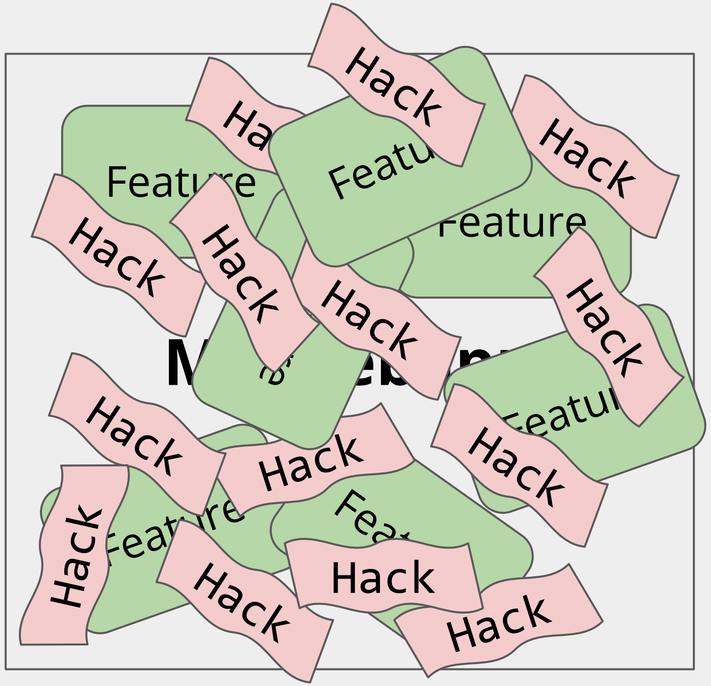

Feature
Hack
Hack
Feature
Hack
My Webapp
Feature
Hack
Feature
Hack

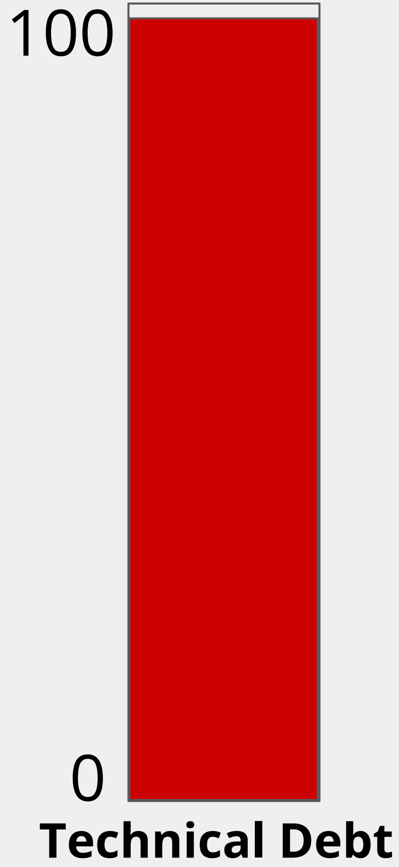

100
0
Technical Debt


Feature
Hack
Feature
Hack
Hack
Hack
Feature
Feature
Hack
Hack
Feature
Feature
Hack
Hack

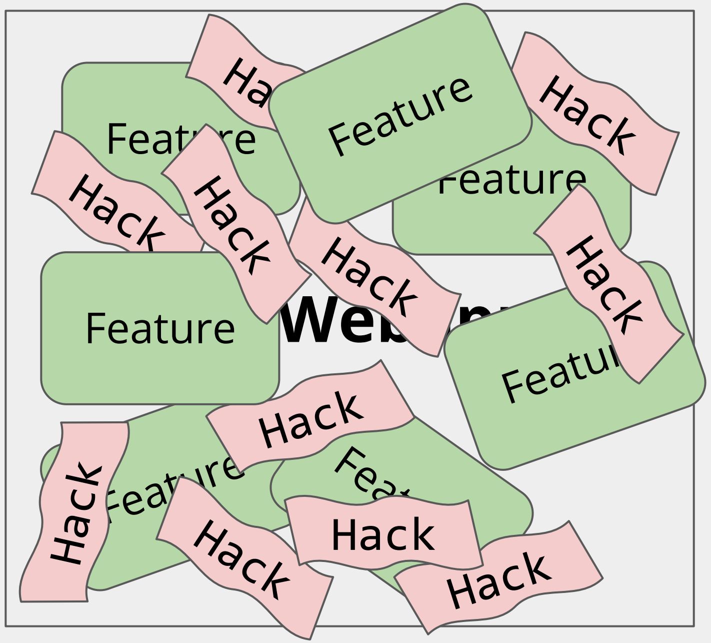

100
0
Technical Debt

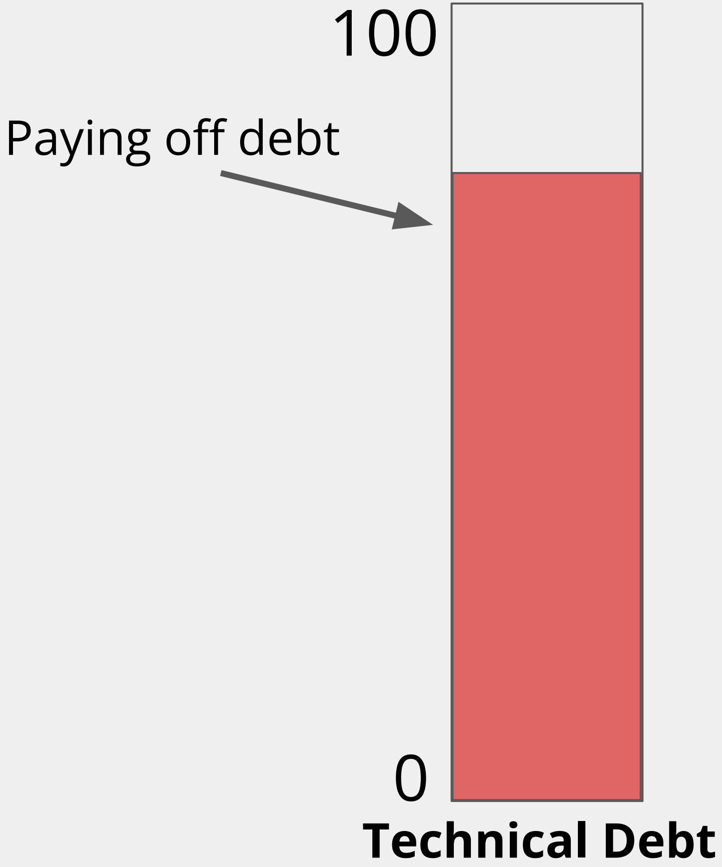
## Code is so bad I can't make any more progress!

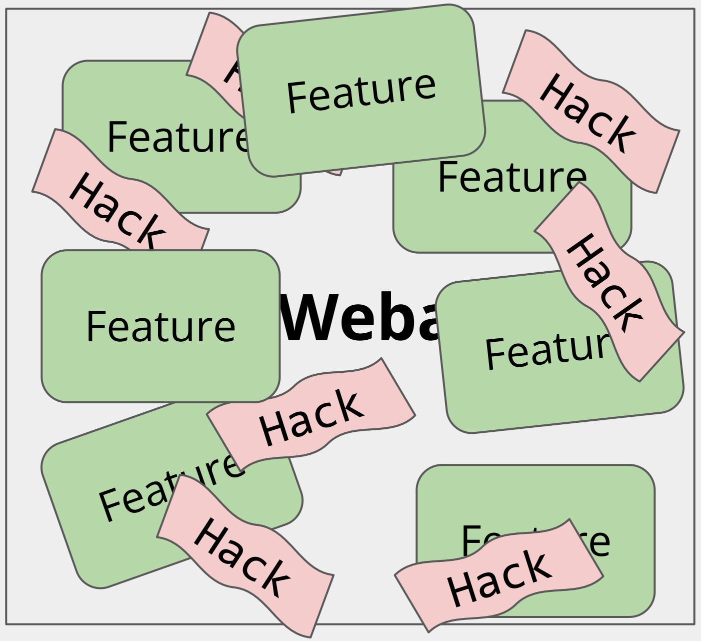


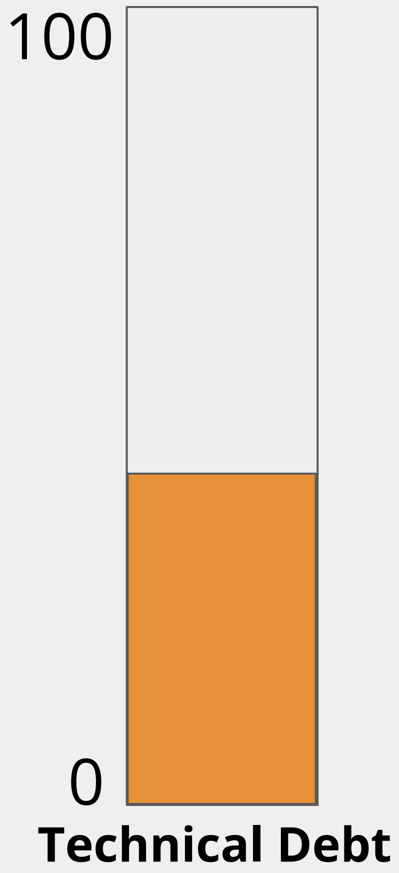


## Have no choice but to refactor...






Feature
Feature
Feature
Feature
Feature
Feature
Feature
Hack
Hack
Hack
Hack
Hack
Hack
Weba


100
0
Technical Debt

## Paying off technical debt

- Accrues **interest** ime
- Would have just e start...

“if you don't have any design, *you don't know what you are making.*”

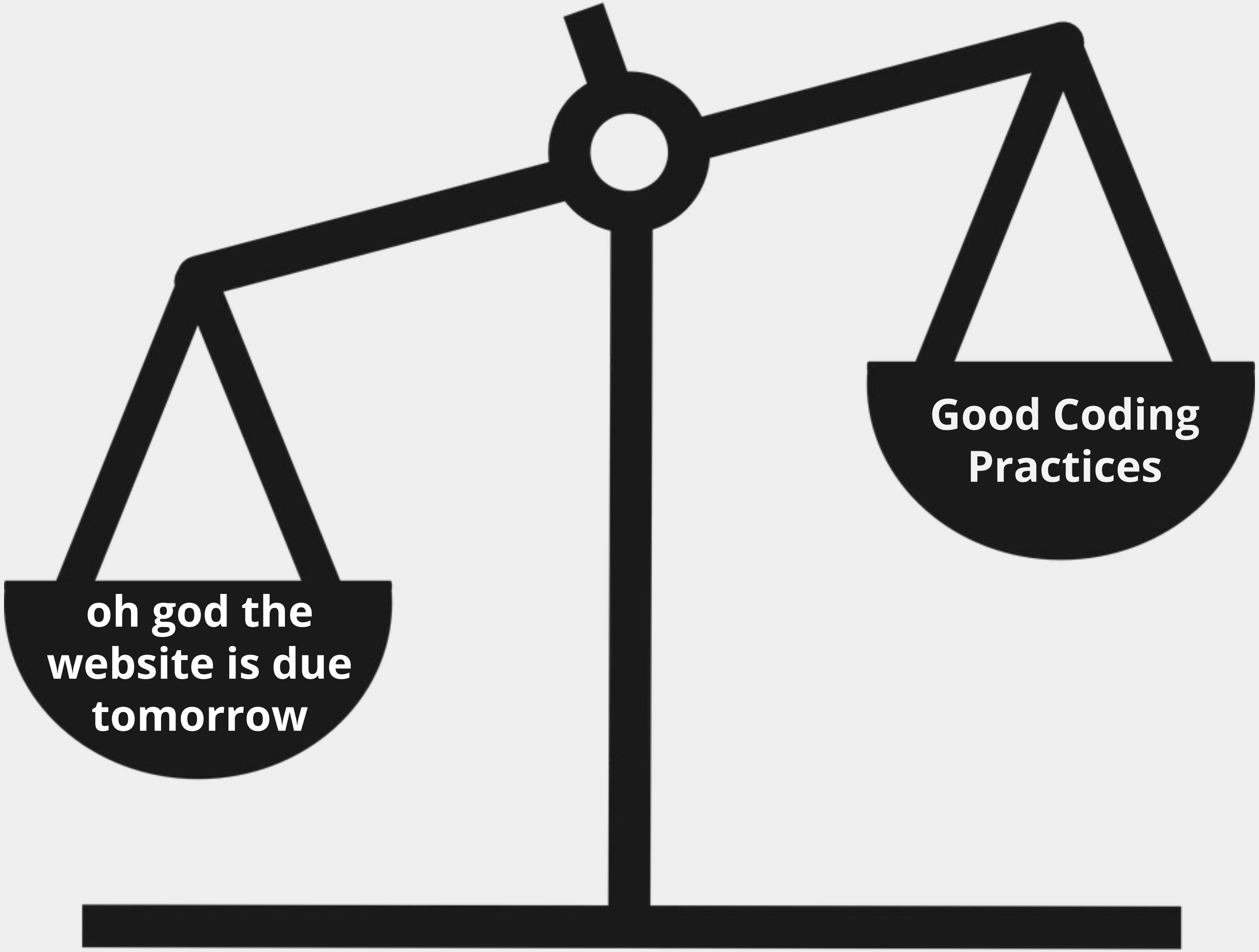
- some dude from stackoverflow

## Testing your code

## it works

- If you don’t test your code, it will eventually break.
- Tests make you feel safe
- Most basic form of testing is UI testing
  - Just open a browser and mess with the UI while you build…it’s not that deep
- Actual Javascript testing (“Jest”): weblab.to/testing
- More on testing:
  - 6.031 😳
  - Even more tests…

## Is it worth it?

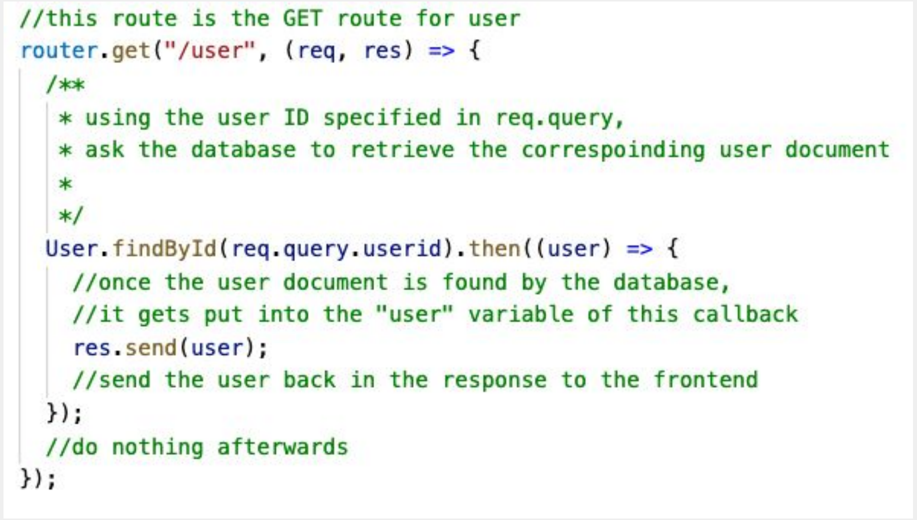

oh god the website is due tomorrow
Good Coding Practices

## No time for good code?

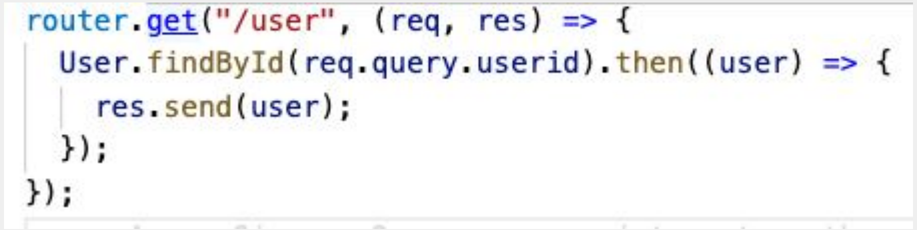
- This class is basically a long hackathon
- You may need some hacky code, but keep it within reason
- Keep your “debt” in check

## Commenting

## Which version is better?

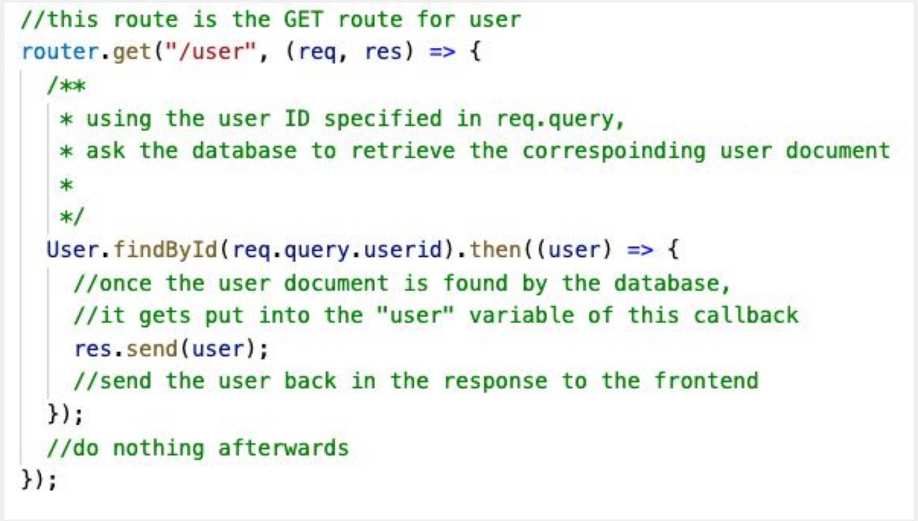


```
//this route is the GET route for user
router.get("/user", (req, res) => {
  /**
   * using the user ID specified in req.query,
   * ask the database to retrieve the correspoinding user document
   *
   */
  User.findById(req.query.userid).then((user) => {
    //once the user document is found by the database,
    //it gets put into the "user" variable of this callback
    res.send(user);
    //send the user back in the response to the frontend
  });
  //do nothing afterwards
});
```

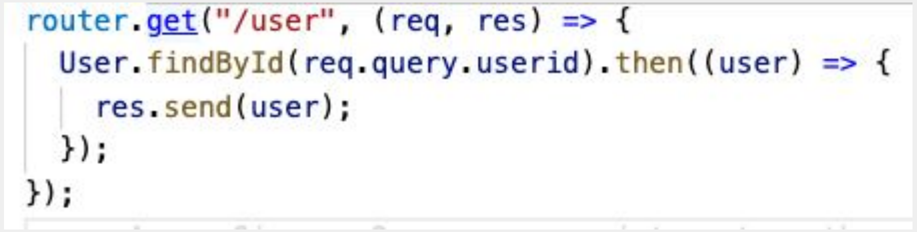


```
router.get("/user", (req, res) => {
  User.findById(req.query.userid).then((user) => {
    res.send(user);
  });
});
```

## Which version is better?



```
//this route is the GET route for user
router.get("/user", (req, res) => {
  /**
   * using the user ID specified in req.query,
   * ask the database to retrieve the correspoinding user document
   *
   */
  User.findById(req.query.userid).then((user) => {
    //once the user document is found by the database,
    //it gets put into the "user" variable of this callback
    res.send(user);
    //send the user back in the response to the frontend
  });
  //do nothing afterwards
});
```



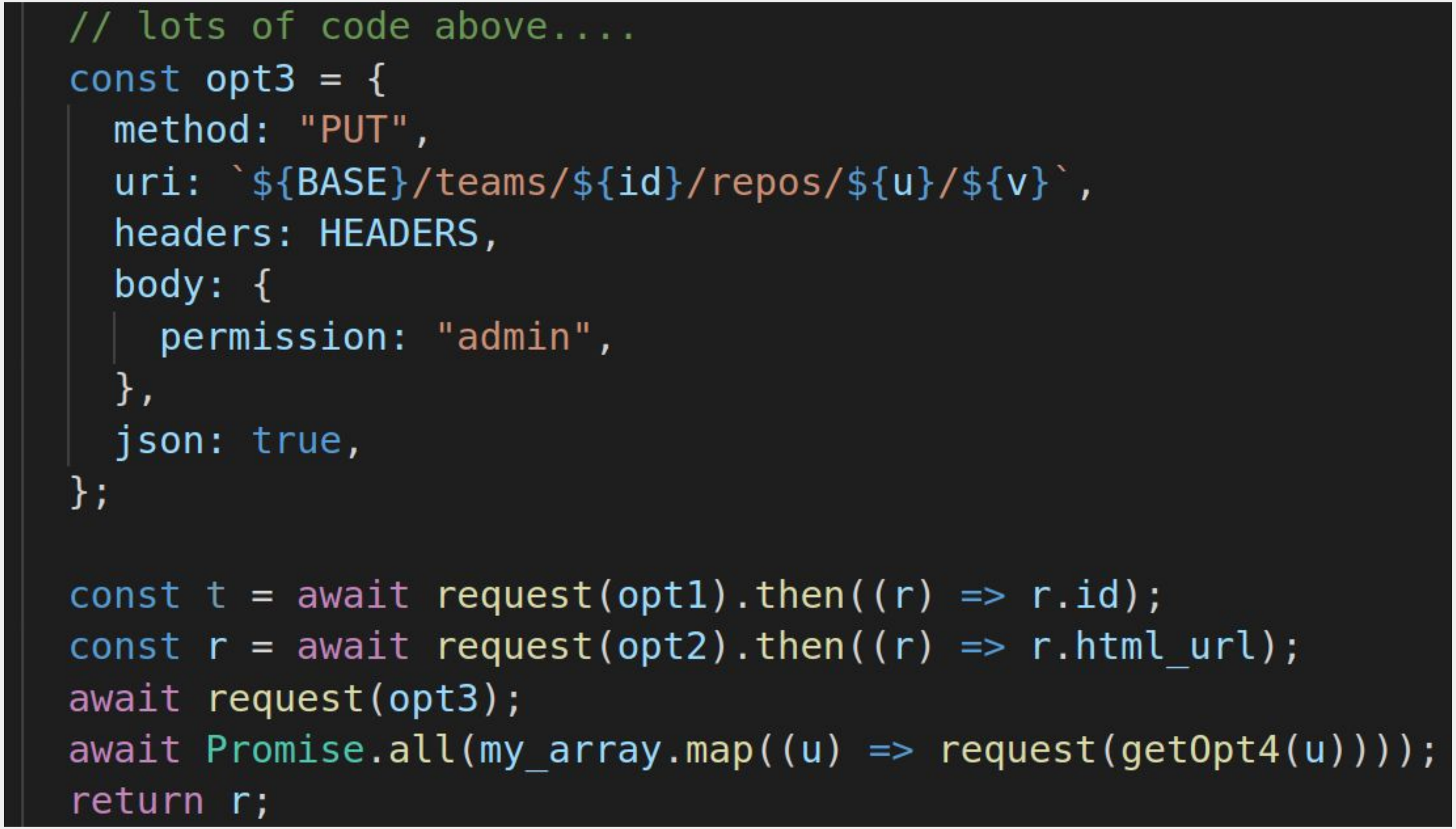
```
router.get("/user", (req, res) => {
  User.findById(req.query.userid).then((user) => {
    res.send(user);
  });
});
```

**I like this one**

## Over-commenting is a thing

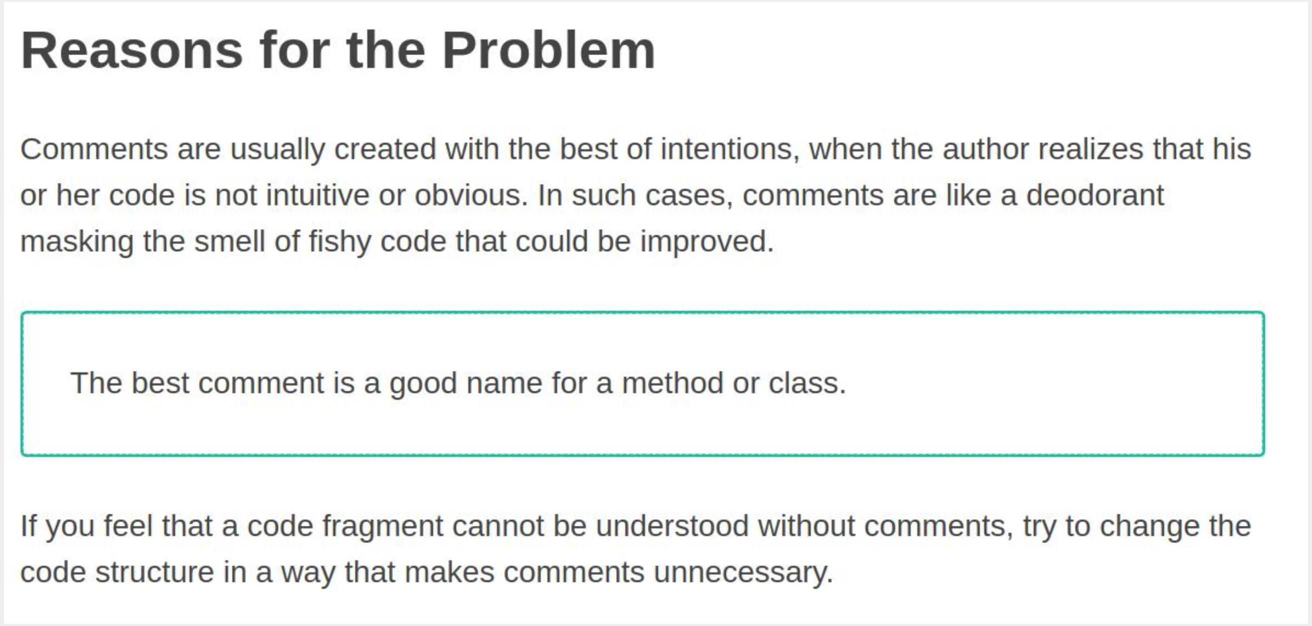
- Don’t use comments that explain the obvious
- Add a comment when you needed to write bad/hacky code
- Add a comment when code is hard to understand…?

# This is hard to understand… should I add comments?



```
// lots of code above....
const opt3 = {
  method: "PUT",
  uri: `${BASE}/teams/${id}/repos/${u}/${v}`,
  headers: HEADERS,
  body: {
    permission: "admin",
  },
  json: true,
};

const t = await request(opt1).then((r) => r.id);
const r = await request(opt2).then((r) => r.html_url);
await request(opt3);
await Promise.all(my_array.map((u) => request(getOpt4(u))));
return r;
```

## This is the wrong time for comments!



### Reasons for the Problem

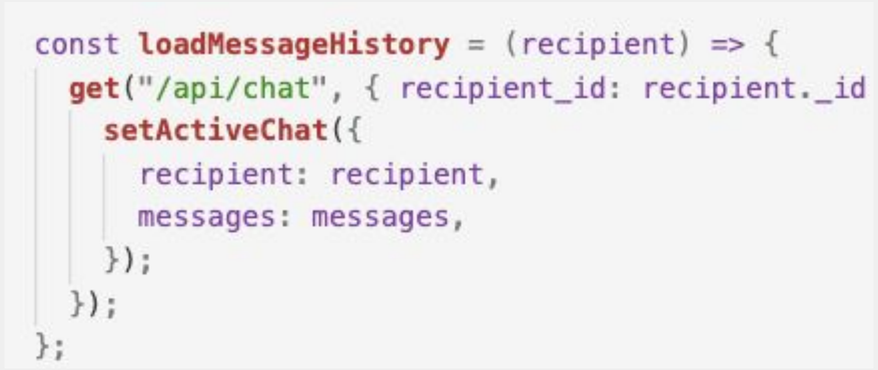
Comments are usually created with the best of intentions, when the author realizes that his or her code is not intuitive or obvious. In such cases, comments are like a deodorant masking the smell of fishy code that could be improved.

> The best comment is a good name for a method or class.

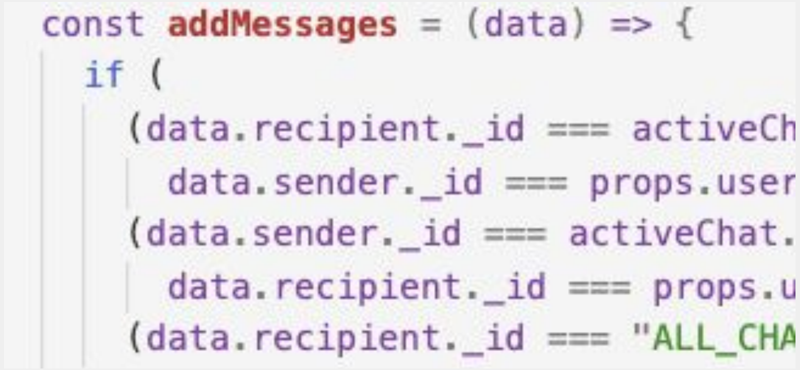
If you feel that a code fragment cannot be understood without comments, try to change the code structure in a way that makes comments unnecessary.

from http://weblab.to/smell

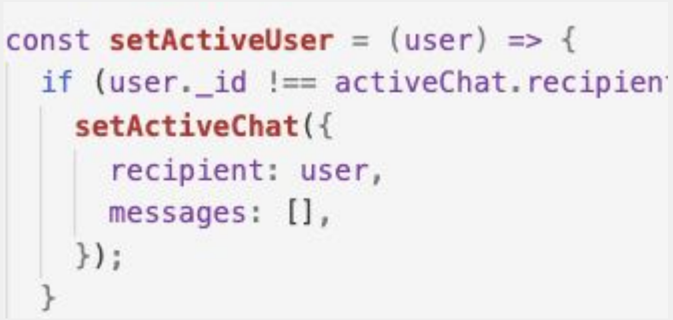
## Write the code using clear function names



```
const loadMessageHistory = (recipient) => {
  get("/api/chat", { recipient_id: recipient._id
    setActiveChat({
      recipient: recipient,
      messages: messages,
    });
  });
};
```

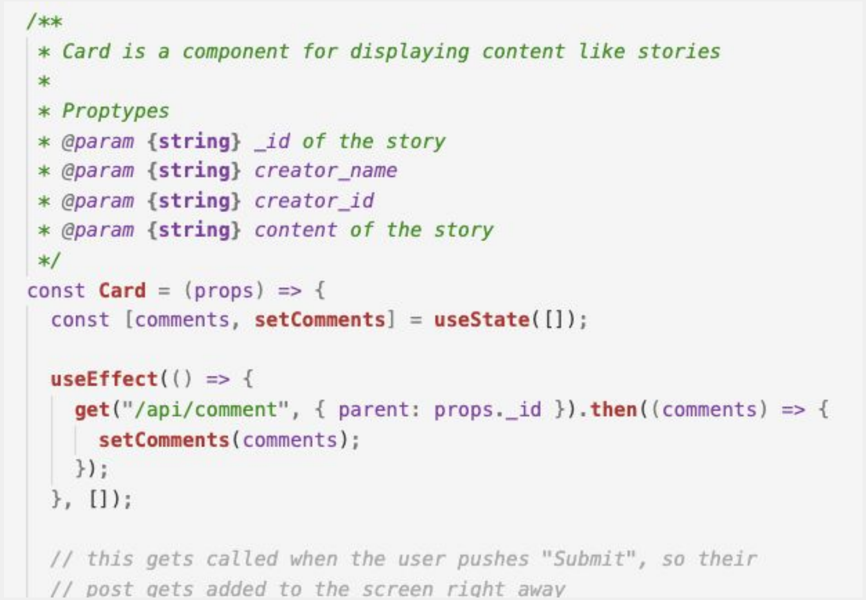


```
const addMessages = (data) => {
  if (
    (data.recipient._id === activeCh
      data.sender._id === props.user
    (data.sender._id === activeChat.
      data.recipient._id === props.u
    (data.recipient._id === "ALL_CHA
```



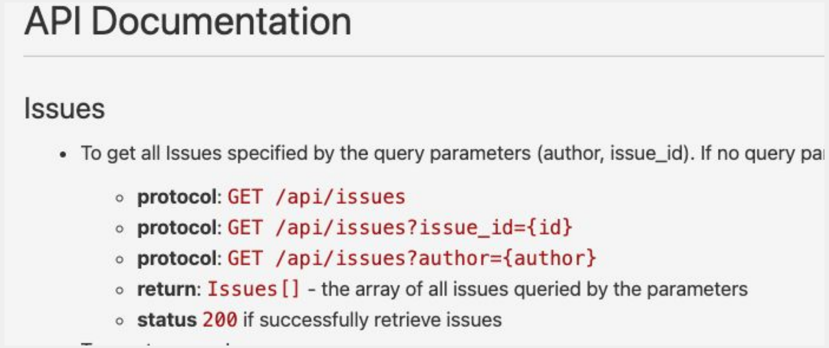
```
const setActiveUser = (user) => {
  if (user._id !== activeChat.recipien
    setActiveChat({
      recipient: user,
      messages: [],
    });
  }
```

# Only comment when “necessary”



```
/**
 * Card is a component for displaying content like stories
 *
 * Proptypes
 * @param {string} _id of the story
 * @param {string} creator_name
 * @param {string} creator_id
 * @param {string} content of the story
 */
const Card = (props) => {
  const [comments, setComments] = useState([]);

  useEffect(() => {
    get("/api/comment", { parent: props._id }).then((comments) => {
      setComments(comments);
    });
  }, []);

  // this gets called when the user pushes "Submit", so their
  // post gets added to the screen right away
```

**JSDocs** (Methods/Objects/Handlers)

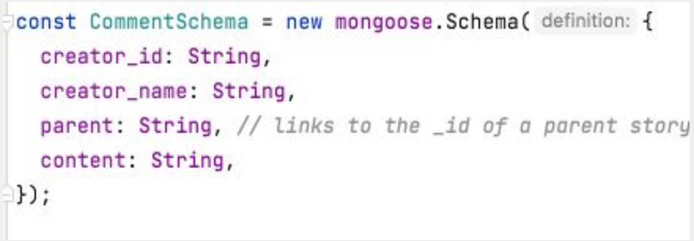


API Documentation

Issues

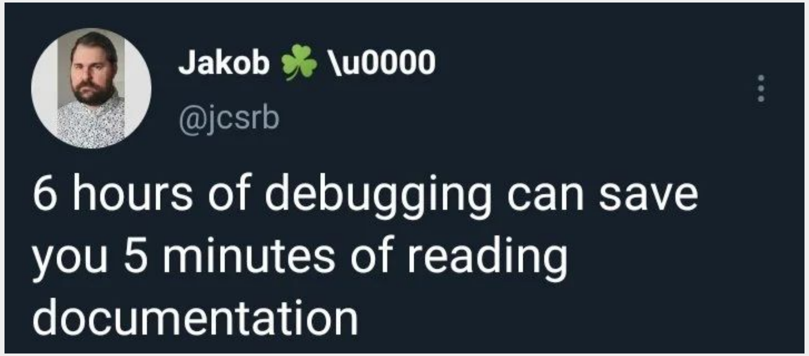
- To get all Issues specified by the query parameters (author, issue_id). If no query pa
  - **protocol**: `GET /api/issues`
  - **protocol**: `GET /api/issues?issue_id={id}`
  - **protocol**: `GET /api/issues?author={author}`
  - **return**: `Issues[]` - the array of all issues queried by the parameters
  - **status** `200` if successfully retrieve issues

**API Routes**



```
const CommentSchema = new mongoose.Schema( definition: {
  creator_id: String,
  creator_name: String,
  parent: String, // links to the _id of a parent story
  content: String,
});
```

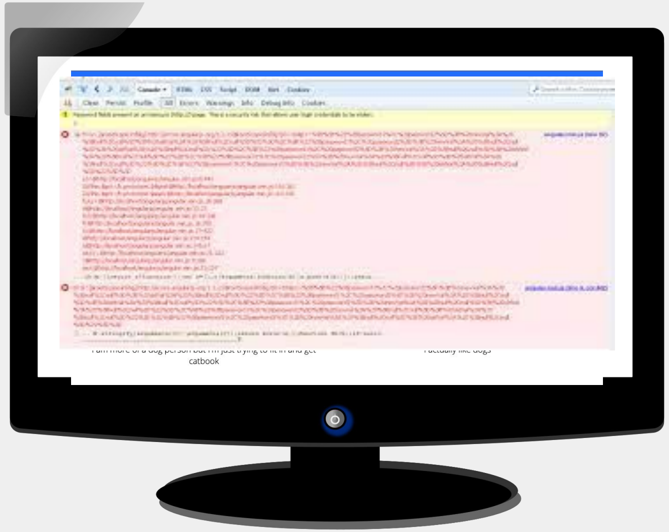
**Database Schemas**


Jakob ☘️ \u0000
@jcsrb

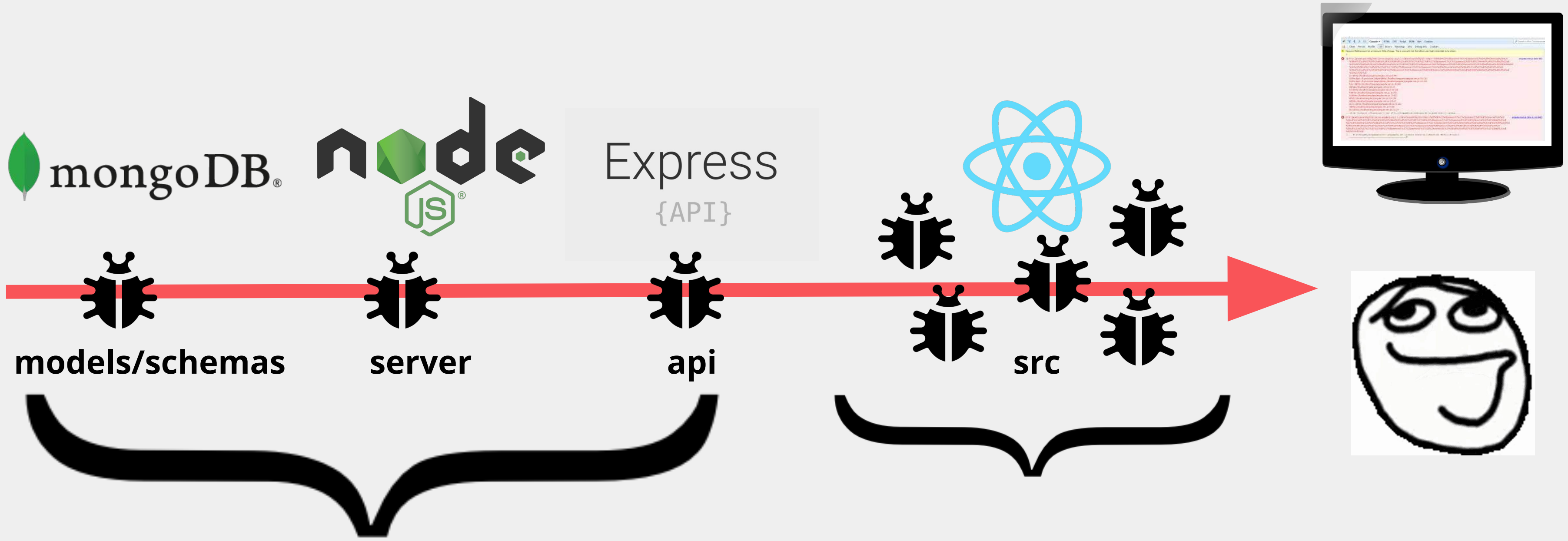
6 hours of debugging can save you 5 minutes of reading documentation

## Debugging

## Where is my bug coming from?



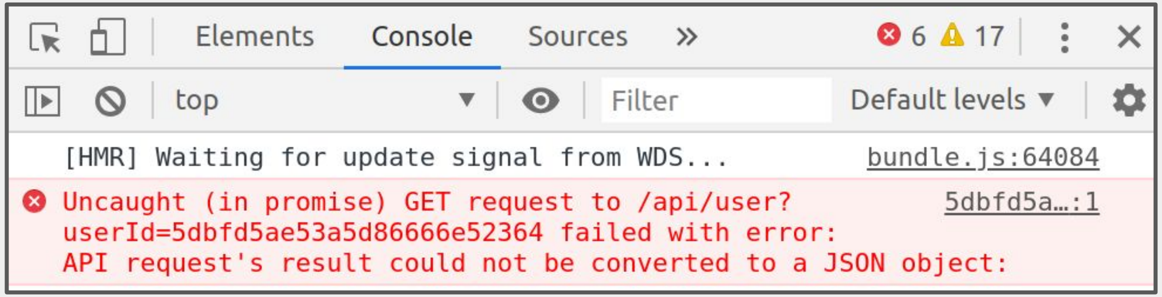
## Where is my bug coming from?



server-side/backend

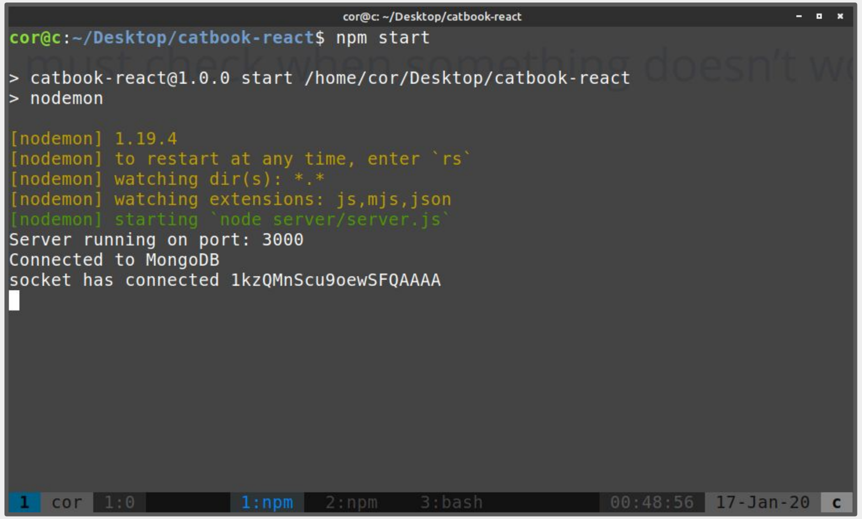
client-side/frontend

# Three things you must check



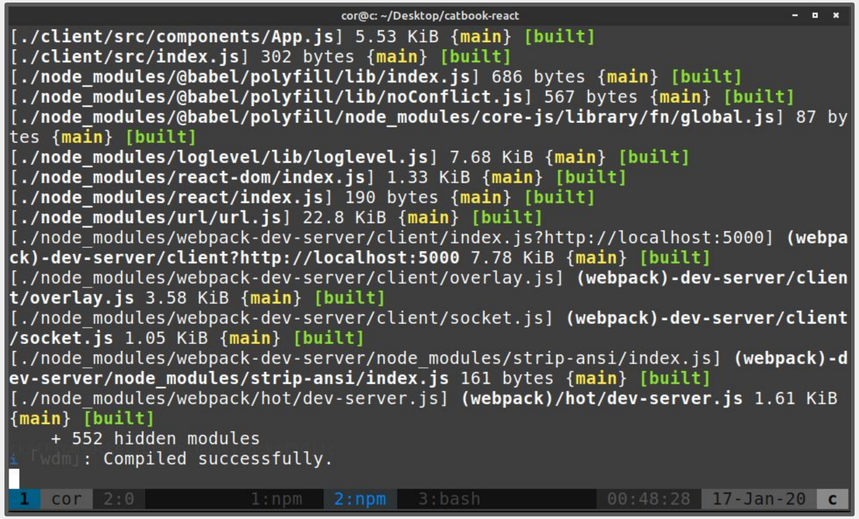
```
[HMR] Waiting for update signal from WDS...                bundle.js:64084
Uncaught (in promise) GET request to /api/user?               5dbfd5a…:1
userId=5dbfd5ae53a5d86666e52364 failed with error:
API request's result could not be converted to a JSON object:
```

Browser console
(F12 or ⌘+⌥+j)



```
cor@c: ~/Desktop/catbook-react
cor@c:~/Desktop/catbook-react$ npm start

> catbook-react@1.0.0 start /home/cor/Desktop/catbook-react
> nodemon

[nodemon] 1.19.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server/server.js`
Server running on port: 3000
Connected to MongoDB
socket has connected 1kzQMnScu9oewSFQAAAA

 1  cor  1:0      1:npm  2:npm    3:bash      00:48:56  17-Jan-20  c
```

npm start

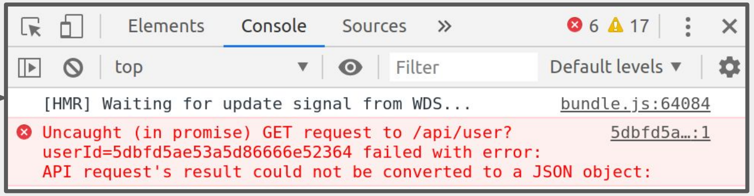


```
cor@c: ~/Desktop/catbook-react
[./client/src/components/App.js] 5.53 KiB {main} [built]
[./client/src/index.js] 302 bytes {main} [built]
[./node_modules/@babel/polyfill/lib/index.js] 686 bytes {main} [built]
[./node_modules/@babel/polyfill/lib/noConflict.js] 567 bytes {main} [built]
[./node_modules/@babel/polyfill/node_modules/core-js/library/fn/global.js] 87 by
tes {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 7.68 KiB {main} [built]
[./node_modules/react-dom/index.js] 1.33 KiB {main} [built]
[./node_modules/react/index.js] 190 bytes {main} [built]
[./node_modules/url/url.js] 22.8 KiB {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost:5000] (webpa
ck)-dev-server/client?http://localhost:5000 7.78 KiB {main} [built]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/clien
t/overlay.js 3.58 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client
/socket.js 1.05 KiB {main} [built]
[./node_modules/webpack-dev-server/node_modules/strip-ansi/index.js] (webpack)-d
ev-server/node_modules/strip-ansi/index.js 161 bytes {main} [built]
[./node_modules/webpack/hot/dev-server.js] (webpack)/hot/dev-server.js 1.61 KiB
{main} [built]
    + 552 hidden modules
i ｢wdm｣: Compiled successfully.

 1  cor  2:0        1:npm  2:npm  3:bash      00:48:28  17-Jan-20  c
```

npm run hotloader

## In React code/frontend

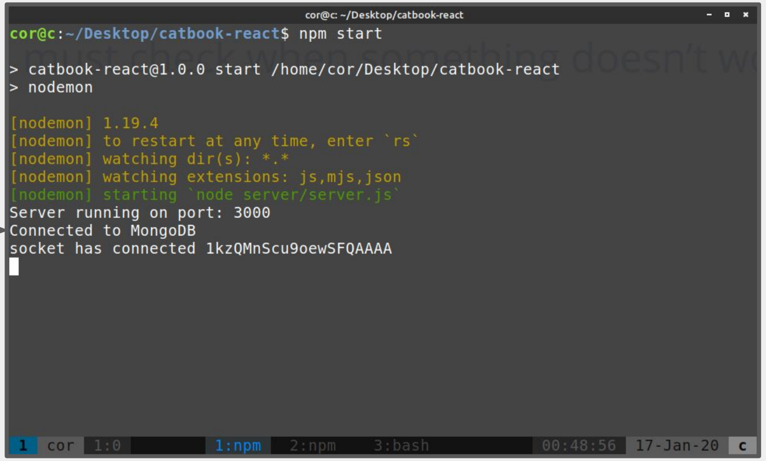
```
console.log("henlo");
```



```
[HMR] Waiting for update signal from WDS...                     bundle.js:64084
Uncaught (in promise) GET request to /api/user?                   5dbfd5a…:1
userId=5dbfd5ae53a5d86666e52364 failed with error:
API request's result could not be converted to a JSON object:
```

## In server code/backend

```
console.log("henlo");
```


cor@c: ~/Desktop/catbook-react

```
cor@c:~/Desktop/catbook-react$ npm start

> catbook-react@1.0.0 start /home/cor/Desktop/catbook-react
> nodemon

[nodemon] 1.19.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server/server.js`
Server running on port: 3000
Connected to MongoDB
socket has connected 1kzQMnScu9oewSFQAAAA
```

1 cor 1:0 1:npm 2:npm 3:bash 00:48:56 17-Jan-20 c

## Finally…Fail Fast!

## Never be in this situation

- I finally finished writing 500 lines of code
- I open up [http://localhost:5000](http://localhost:5000) to test my code and...
- Nothing works! What to do?

**Option A:**

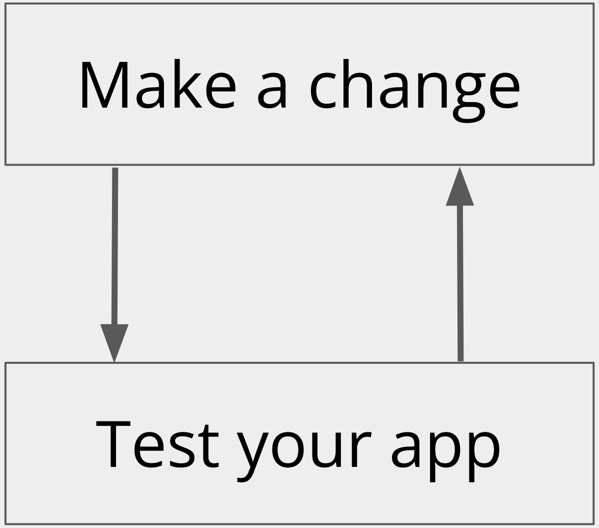
**Option B:**

More lines of code = More to debug

- conservation of bugs

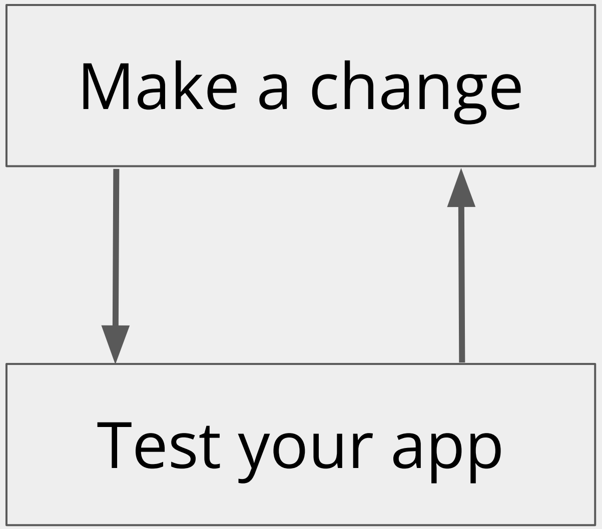
## A short development/debug loop

- Make **small, incremental** changes
- Keep functions short and modular
- Test at every opportunity (e.g. `console.log` and make sure it looks right)



## Optimizing the debug loop

- Shaving off a few seconds per iteration adds up
- Tools to optimize the debug loop: react hotloader, nodemon



## Debugging thoroughly

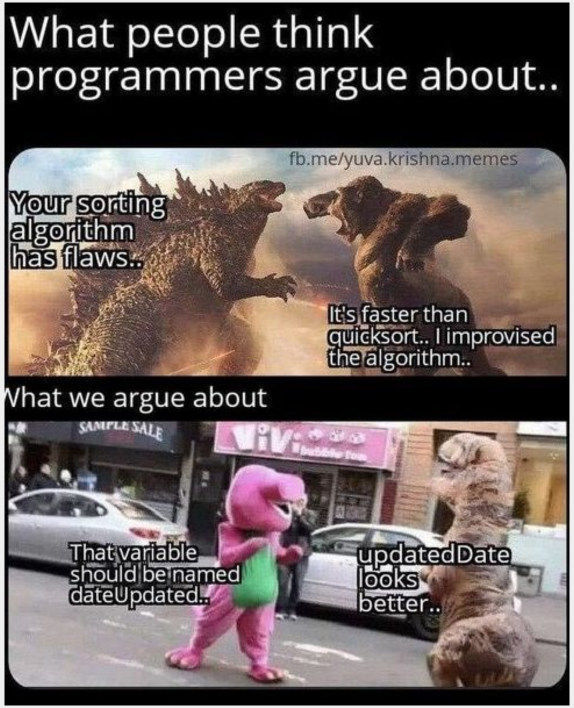
- Even if your code *seems* to work, there may still be bugs…
- Users may break your app in ways you never imagined

Linus’s law:
"given enough eyeballs, all bugs are shallow".

- All team members should thoroughly try to break the site
- Get a friend to test your website
- **A stable, simple website beats a buggy, complex website**

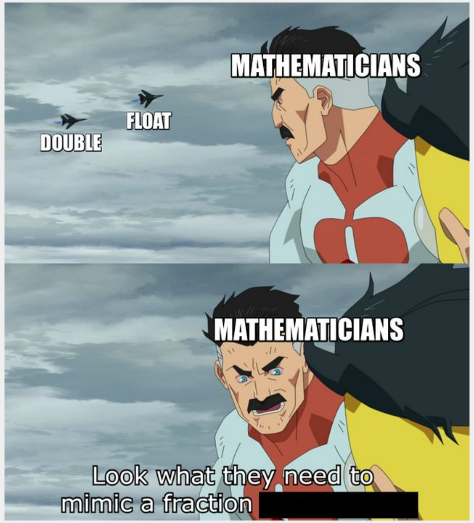
## Summary

1. git responsibly
2. Limit hacky code
3. Write code that doesn't need lots of comments
4. Develop with a tight debug loop
5. Slow down and fix your bugs


What people think programmers argue about..
fb.me/yuva.krishna.memes
Your sorting algorithm has flaws..
It's faster than quicksort.. I improvised the algorithm..
What we argue about
That variable should be named dateUpdated..
updatedDate looks better..

## More Memes

git
I prefer the real version control
SUBVERSION
I said the *real* version control
Final 2
Final Final
FINAL FOR REAL THIS TIME
OK This is definitely final
Perfection

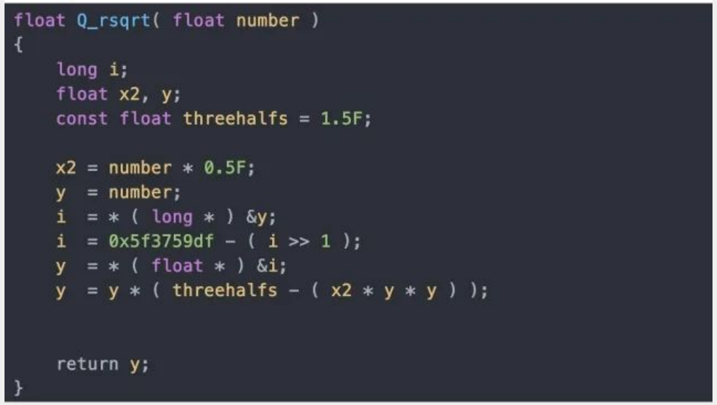


my program: *works perfectly*

me: *cleans up the code*

also my program:

Company : Has a very strict "No comment" policy, because comments = bad code.
Companies code :



```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y  = number;
    i  = * ( long * ) &y;
    i  = 0x5f3759df - ( i >> 1 );
    y  = * ( float * ) &i;
    y  = y * ( threehalfs - ( x2 * y * y ) );


    return y;
}
```