ZenHub

Functional Programming with Ramda

Christine Legge

http://bit.ly/fp-with-ramda

ZenHub

Hello!

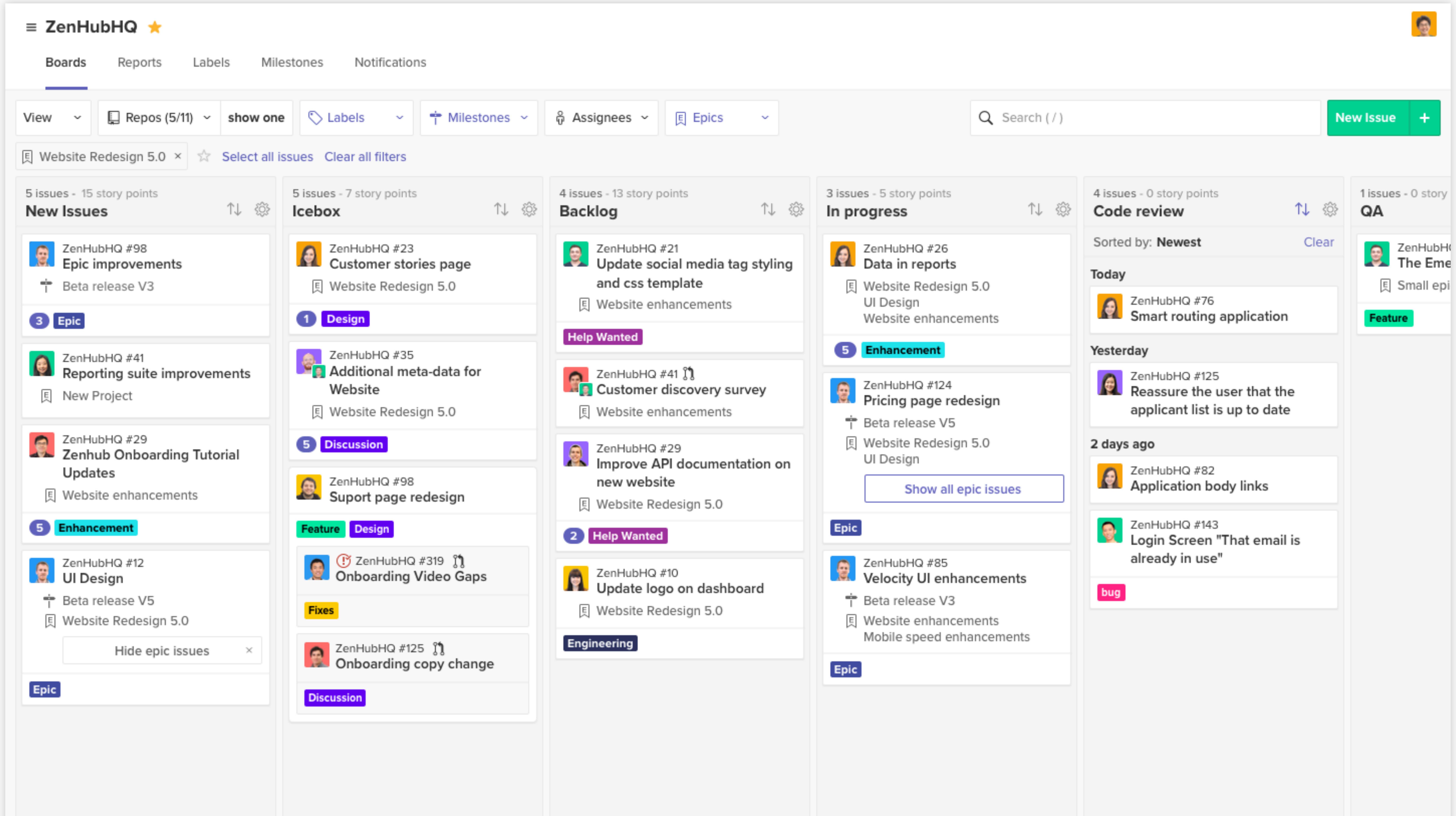
–

Christine Legge

Software Engineer

http://bit.ly/fp-with-ramda

ZenHub



ZenHubHQ

Boards Reports Labels Milestones Notifications

View | Repos (5/11) | show one | Labels | Milestones | Assignees | Epics | Search (/) | New Issue +

Website Redesign 5.0 × | Select all issues | Clear all filters

5 issues - 15 story points
New Issues

- ZenHubHQ #98 Epic improvements — Beta release V3 — 3 Epic
- ZenHubHQ #41 Reporting suite improvements — New Project
- ZenHubHQ #29 Zenhub Onboarding Tutorial Updates — Website enhancements — 5 Enhancement
- ZenHubHQ #12 UI Design — Beta release V5 — Website Redesign 5.0 — Hide epic issues × — Epic

5 issues - 7 story points
Icebox

- ZenHubHQ #23 Customer stories page — Website Redesign 5.0 — 1 Design
- ZenHubHQ #35 Additional meta-data for Website — Website Redesign 5.0 — 5 Discussion
- ZenHubHQ #98 Suport page redesign — Feature Design
  - ZenHubHQ #319 Onboarding Video Gaps — Fixes
  - ZenHubHQ #125 Onboarding copy change — Discussion

4 issues - 13 story points
Backlog

- ZenHubHQ #21 Update social media tag styling and css template — Website enhancements — Help Wanted
- ZenHubHQ #41 Customer discovery survey — Website enhancements
- ZenHubHQ #29 Improve API documentation on new website — Website Redesign 5.0 — 2 Help Wanted
- ZenHubHQ #10 Update logo on dashboard — Website Redesign 5.0 — Engineering

3 issues - 5 story points
In progress

- ZenHubHQ #26 Data in reports — Website Redesign 5.0, UI Design, Website enhancements — 5 Enhancement
- ZenHubHQ #124 Pricing page redesign — Beta release V5 — Website Redesign 5.0, UI Design — Show all epic issues — Epic
- ZenHubHQ #85 Velocity UI enhancements — Beta release V3 — Website enhancements, Mobile speed enhancements — Epic

4 issues - 0 story points
Code review

Sorted by: Newest Clear

Today
- ZenHubHQ #76 Smart routing application

Yesterday
- ZenHubHQ #125 Reassure the user that the applicant list is up to date

2 days ago
- ZenHubHQ #82 Application body links
- ZenHubHQ #143 Login Screen "That email is already in use" — bug

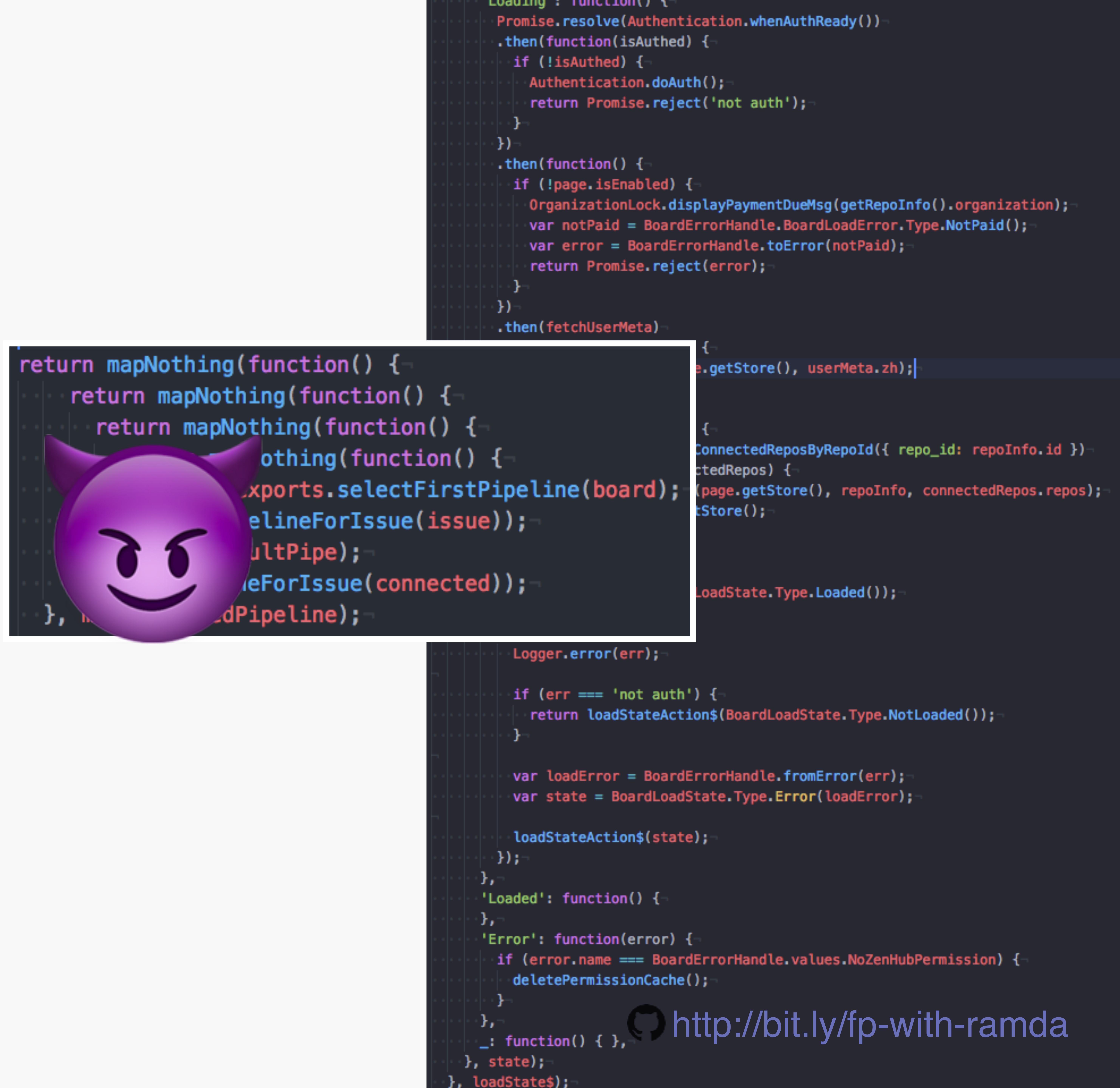
1 issues - 0 story
QA

- ZenHubH The Eme — Small epi — Feature

ZenHub

THE VILLAIN

- unreadable
- complex
- hard to reason about
- hard to test
- BUGS



```
      Promise.resolve(Authentication.whenAuthReady())
      .then(function(isAuthed) {
        if (!isAuthed) {
          Authentication.doAuth();
          return Promise.reject('not auth');
        }
      })
      .then(function() {
        if (!page.isEnabled) {
          OrganizationLock.displayPaymentDueMsg(getRepoInfo().organization);
          var notPaid = BoardErrorHandle.BoardLoadError.Type.NotPaid();
          var error = BoardErrorHandle.toError(notPaid);
          return Promise.reject(error);
        }
      })
      .then(fetchUserMeta)
                                          {
                                          e.getStore(), userMeta.zh);

                                          {
                                          ConnectedReposByRepoId({ repo_id: repoInfo.id })
                                          ctedRepos) {
                                          (page.getStore(), repoInfo, connectedRepos.repos);
                                          tStore();

                                          LoadState.Type.Loaded());

        Logger.error(err);

        if (err === 'not auth') {
          return loadStateAction$(BoardLoadState.Type.NotLoaded());
        }

        var loadError = BoardErrorHandle.fromError(err);
        var state = BoardLoadState.Type.Error(loadError);

        loadStateAction$(state);
      });
    },
    'Loaded': function() {
    },
    'Error': function(error) {
      if (error.name === BoardErrorHandle.values.NoZenHubPermission) {
        deletePermissionCache();
      }
    },
    _: function() { },
  }, state);
}, loadState$);
```

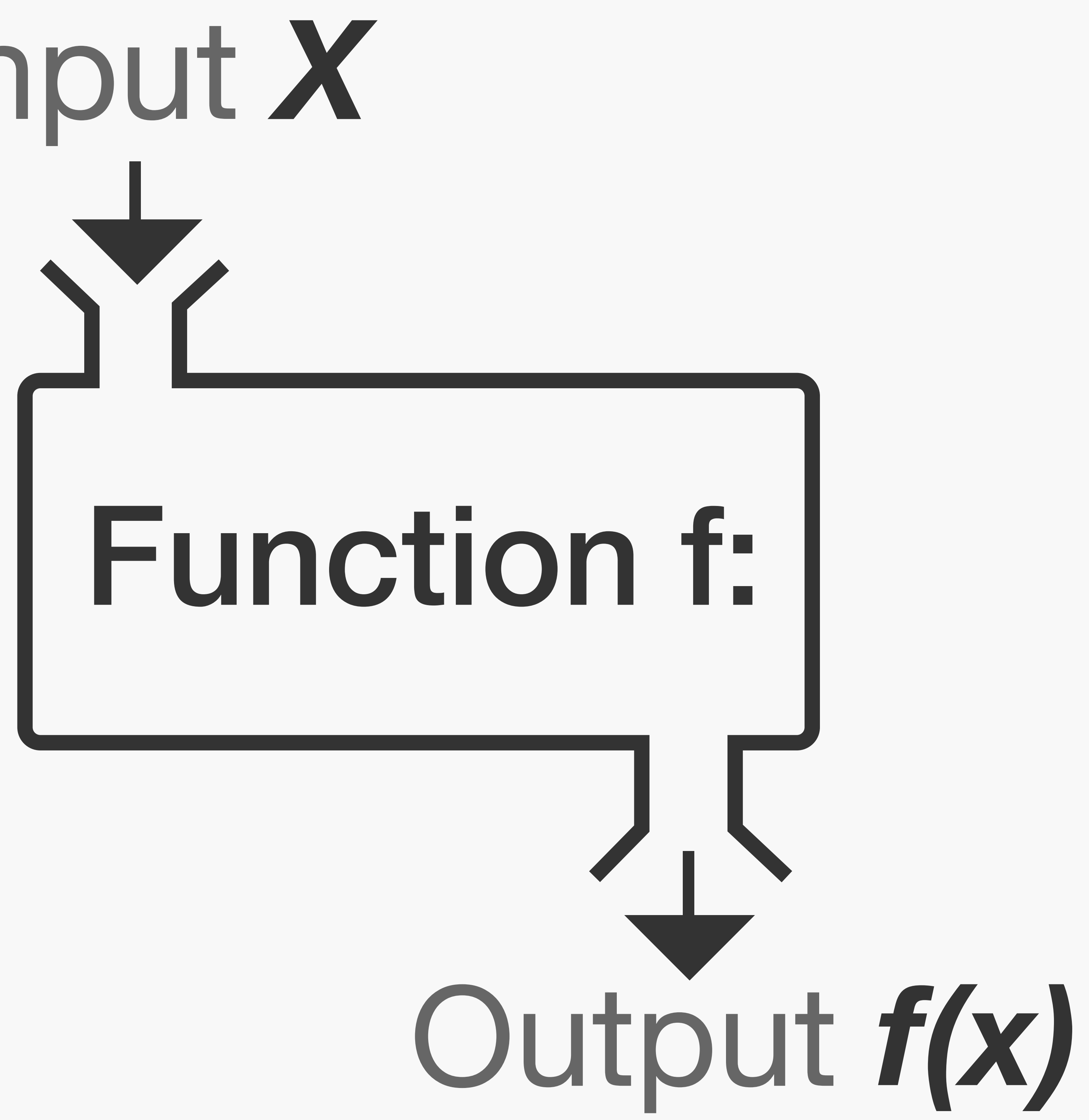
```
return mapNothing(function() {
    return mapNothing(function() {
      return mapNothing(function() {
                      othing(function() {
                     xports.selectFirstPipeline(board);
                     elineForIssue(issue));
                     ultPipe);
                     eForIssue(connected));
  },                 dPipeline);
```

http://bit.ly/fp-with-ramda

ZenHub

THE HERO

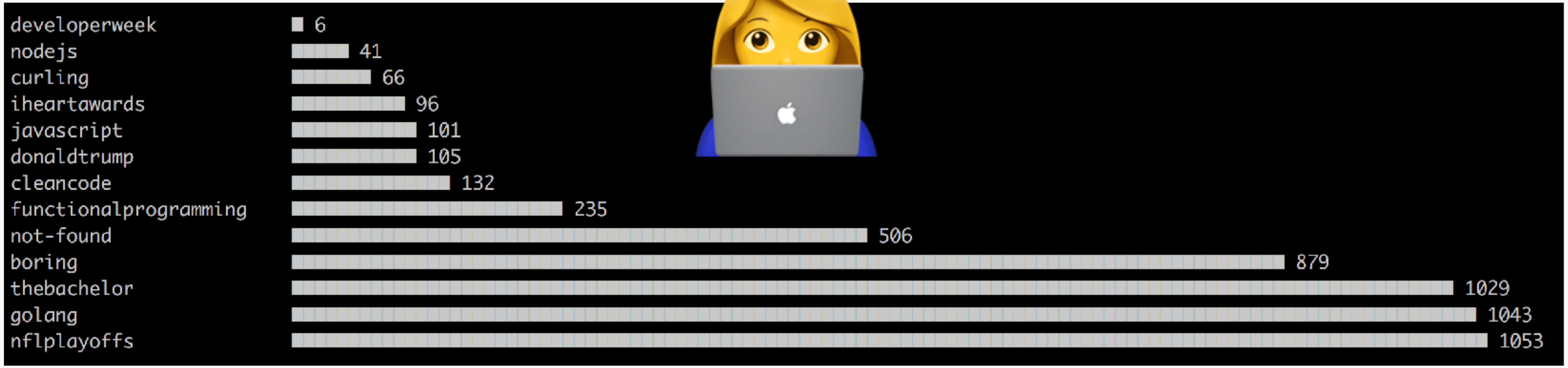
Functional Programming with Ramda



http://bit.ly/fp-with-ramda

ZenHub

THE GOAL



```
developerweek          ▇ 6
nodejs                 ▇▇▇▇▇ 41
curling                ▇▇▇▇▇▇▇ 66
iheartawards           ▇▇▇▇▇▇▇▇▇▇ 96
javascript             ▇▇▇▇▇▇▇▇▇▇▇ 101
donaldtrump            ▇▇▇▇▇▇▇▇▇▇▇ 105
cleancode              ▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 132
functionalprogramming  ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 235
not-found              ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 506
boring                 ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 879
thebachelor            ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 1029
golang                 ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 1043
nflplayoffs            ▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇▇ 1053
```

```
const getHashtagTweetCountStrings = R.pipe(
  removeDuplicates
  groupByHashtag,
  getTweetCountForHashtags,
  getGraphStrings('█', 10),
);
```

http://bit.ly/fp-with-ramda

WHAT IS FUNCTIONAL PROGRAMMING?

- Programming paradigm
- Coding style

WHAT IS RAMDA?

- JavaScript utility library
- functional approach

ZenHub

TODAY

Functional Programming:

- Immutability
- Referential Transparency
- First Class functions

Ramda Features:

- Automatic currying
- Function first, data last API

Function Composition

Analyze some Tweets

ZenHub

IMMUTABILITY

- Objects cannot be modified after they are created
- Ramda functions never mutate input data

ZenHub

```
const tweet = {
  text: 'Welcome to DeveloperWeek!',
};

// not functional
tweet.user = 'christine';  😈

// functional
R.assoc('favourites', 1000, tweet);  👩‍💻
```

ZenHub

REFERENTIAL TRANSPARENCY

- Pure functions(same input -> same output)
- Side effect free functions (don’t modify state outside scope)
- Ramda functions are all referentially transparent

ZenHub

```
const list = [1, 2, 3, 4, 5];

// not pure
list.reverse(); 😈

// pure
R.reverse(list); 👩‍💻
```

```
let counter = 0;

// has side effects
function incrementSideEffects() {
  counter++;                                  😈
  return counter;
}

// no side effects
function incrementNoSideEffects(num) {  👩‍💻
  return num + 1;
}
```

ZenHub

FIRST CLASS FUNCTIONS

- Assign functions to variables
- Pass functions as arguments
- Return functions from functions

ZenHub

```
// Functions can be assigned to variables
const abs = Math.abs;
abs(-1) // => 1


// Functions can be passed as parameters
function map(fn, array) {
  return array.map(fn);
}
map(abs, [-1, 2, -3]); // => [1, 2, 3]
```

ZenHub

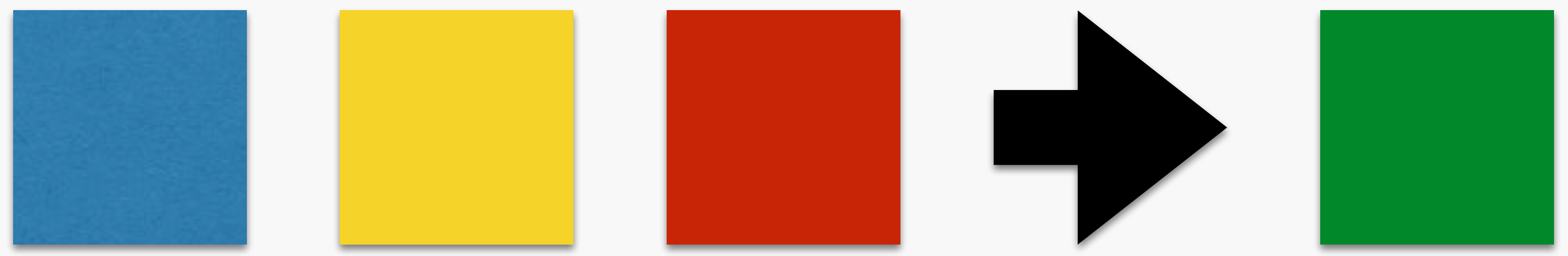
CURRYING

```
// Regular implementation
function add(a, b) {
  return a + b;
}

// Curried function
const curriedAdd = R.curry(add);
```

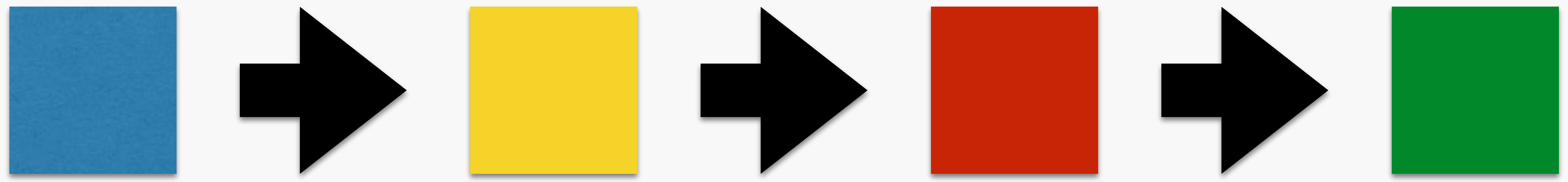
```
add(1); // => NaN 😈
add(1, 2); // => 3

curriedAdd(1); // => [Function]
curriedAdd(1)(2); // => 3

const add5 = curriedAdd(5);
add5(7); // => 12
add5(3); // => 8
```

👩‍💻

http://bit.ly/fp-with-ramda



A curried function can be called with a subset of its parameters and it will return a function that expects the remaining parameters



ZenHub

WHY RAMDA?

ZenHub

TODAY

Functional Programming:

- Immutability
- Referential Transparency
- First Class functions

Ramda Features:

- Automatic currying
- Function first, data last API

Function Composition

Analyze some Tweets

ZenHub

AUTOMATIC CURRYING

- All functions are automatically curried

ZenHub

```
const tweet1 = {
  text: 'Just setting up my Twitter...',
  user: 'christine',
  favourites: 1
};

const tweetList = [
  {
    text: '...Best photo ever. #oscars',
    user: 'ellen',
    favourites: 2395667
  },
  ...
];
```

```
R.prop('user', tweet1); // => 'christine'
const getUser = R.prop('user');
getUser(tweet1); // => 'christine'
```

ZenHub

FUNCTION FIRST API

- The reverse of native JS and similar libraries
- All functions take the data as the last parameter

ZenHub

```
const isPopular = tweet => tweet.favourites > 500000;

const filterPopularTweets = tweets => (
  _.filter(tweets, isPopular)
);
_.filter(tweetList1, isPopular);
_.filter(tweetList2, isPopular);

const filterPopularTweets = R.filter(isPopular);
filterPopularTweets(tweetList1);
filterPopularTweets(tweetList2)
```

😈

👩‍💻

ZenHub

WHY IS THIS IMPORTANT?

ZenHub

FUNCTION COMPOSITION 👩‍💻

- The process of passing the result of one function to the input of the next function
- chaining calls of functions
- combine simple building blocks

“The problem is that you can’t avoid composition just because you’re not aware of it. You still do it —but you do it badly.”

Eric Elliott

ZenHub

```
formatString('Christine Legge') // => 'christine-legge'
```

1. Split the name on spaces
2. Map the strings to lower case
3. Join the strings with dashes
4. Encode the URI component

```
// Plain javascript with intermediate variables
function formatString(input) {
  const splitString = input.split(' ');
  const lowerCaseString = splitString.map(str => str.toLowerCase());
  const joinedString = lowerCaseString.join('-');
  return encodeURIComponent(joinedString);
}
```

ZenHub

```
const formatString = R.pipe(
  R.split(' '),       // String -> [String]
  R.map(R.toLower),   // [String] -> [String]
  R.join('-'),        // [String] -> String
  encodeURIComponent // String -> String
);

formatString('Christine Legge’) // => 'christine-legge'
```

1. Split the name on spaces
2. Map the strings to lower case
3. Join the strings with dashes
4. Encode the URI component

http://bit.ly/fp-with-ramda

ZenHub

```
// Plain javascript
const formatString = input => encodeURIComponent(
  input.split(' ')
    .map(str => str.toLowerCase())
    .join('-')
);
```

ZenHub

Let’s look at some Tweets!

http://bit.ly/fp-with-ramda

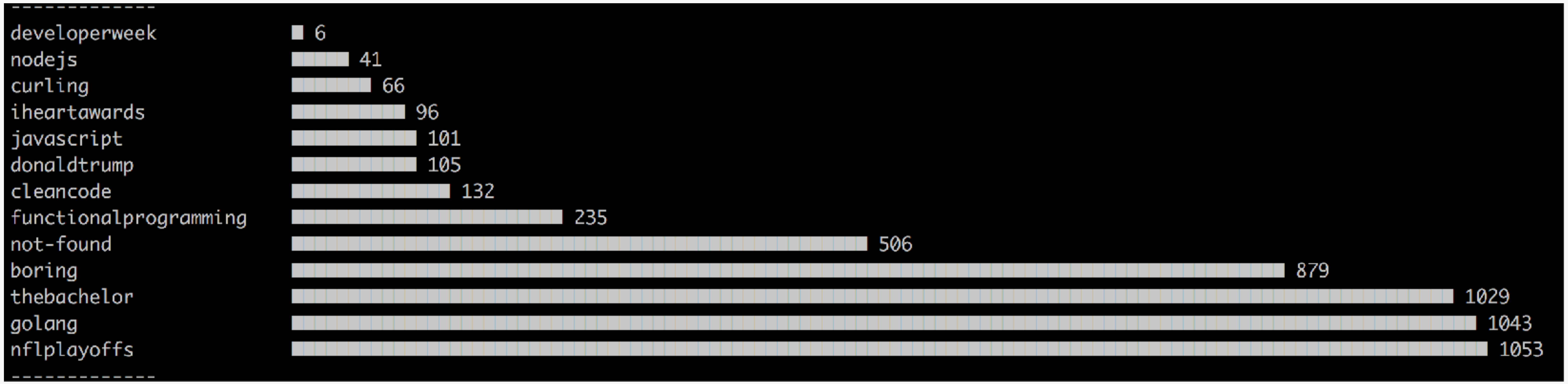
ZenHub

```
[
  {
    "created_at": "Tue Jan 16 13:33:37 +0000 2018",
    "id": 953258943123476500,
    "id_str": "953258943123476480",
    "text": "If only Bradley’s arm was longer. Best photo ever. #oscars",
    "entities": {
      "hashtags": [
        {
          "text": “oscars",
        }
      ],
    "favorite_count": 2395667,
  },
  {
    "created_at": "Tue Jan 09 15:54:28 +0000 2018",
   "id": 950757675117174800,
    "id_str": "950757675117174784",
    "text": "Just setting up my Twitter. #myfirstTweet",
    "entities": {
      "hashtags": [
        {
          "text": "myfirstTweet",
        }
      ],
    "favorite_count": 1,
  }
]
```

http://bit.ly/fp-with-ramda

ZenHub

`getTweetCountGraph(tweets)`



```
-------------
developerweek                █ 6
nodejs                       █████ 41
curling                      ███████ 66
iheartawards                 ██████████ 96
javascript                   ███████████ 101
donaldtrump                  ███████████ 105
cleancode                    ██████████████ 132
functionalprogramming        ████████████████████████ 235
not-found                    ███████████████████████████████████████████████████ 506
boring                       ████████████████████████████████████████████████████████████████████████████████████████ 879
thebachelor                  ███████████████████████████████████████████████████████████████████████████████████████████████████████ 1029
golang                       █████████████████████████████████████████████████████████████████████████████████████████████████████████ 1043
nflplayoffs                  ██████████████████████████████████████████████████████████████████████████████████████████████████████████ 1053
-------------
```

http://bit.ly/fp-with-ramda

ZenHub

1. Clean data
2. Group tweets by hashtags
3. Count the number of tweets per hashtag
4. Construct a graph

ZenHub

1. Clean data

```
{
  "created_at": "Tue Jan 16 13:33:37 +0000 2018",
  "id": 953258943123476500,
  "id_str": "953258943123476480",
  "text": "If only Bradley’s arm was longer. Best photo ever. #oscars",
  "entities": {
    "hashtags": [
      {
        "text": “oscars",
      }
    ],
  "favorite_count": 2395667,
}
```

ZenHub

2. Group tweets by hashtags

```
{
  "created_at": "Tue Jan 16 13:33:37 +0000 2018",
  "id": 953258943123476500,
  "id_str": "953258943123476480",
  "text": "If only Bradley’s arm was longer. Best photo ever. #oscars",
  "entities": {
    "hashtags": [
      {
        "text": "oscars",
      }
    ],
  "favorite_count": 2395667,
}
```

ZenHub

3. Count the number of tweets per hashtag

```
{
  "developerweek": [
    {
      "created_at": "Tue Jan 16 13:33:37 +0000 2018",
      "id": 953258943123476500,
      "id_str": "953258943123476480",
      "text": "Come learn about RAMDA! #DeveloperWeek 2018"
    },
    ...
  ],
  "curling": [
    {
      "created_at": "Tue Jan 16 04:10:54 +0000 2018",
      "id": 953117331366047700,
      "id_str": "953117331366047744",
      "text": "#Curling in #Kosovo? This #Canadian wants to make it happen"
    }
    ...
  ]
}
```

ZenHub

4. Construct a graph

```
{
  "boring": 879,
  "not-found": 515,
  "thebachelor": 1029,
  "nflplayoffs": 1053,
  "donaldtrump": 318,
  "cleancode": 132,
  "javascript": 105,
  "curling": 66,
  "developerweek": 6,
  "functionalprogramming": 236,
  "golang": 1043,
  "nodejs": 49,
  "iheartawards": 97
}
```

ZenHub

FUNCTIONAL PROGRAMMING

- Start writing in a functional style today

ZenHub

FUNCTION COMPOSITION

- You’re probably already doing it

ZenHub

RAMDA

- Help you write JavaScript in a functional style
- Makes function composition simple

ZenHub

http://bit.ly/fp-with-ramda

ZenHub

Thank you! Questions?

http://bit.ly/fp-with-ramda