PROJECT HUB

WEEK 2

BRINGING AN IDEA TO LIFE

ICICS 014

3:30PM | FEB 16 | BIWEEKLY

UBC LAUNCH PAD

SOFTWARE ENGINEERING TEAM

UBC CSSS

PROJECT HUB

UBC LAUNCH PAD
SOFTWARE ENGINEERING TEAM

UBC CSSS

- Biweekly (hopefully)
  - Workshop or talk
  - Demos!
  - Open workspace
- Food!
- Make friends!
- Learn stuff!

PROJECT HUB

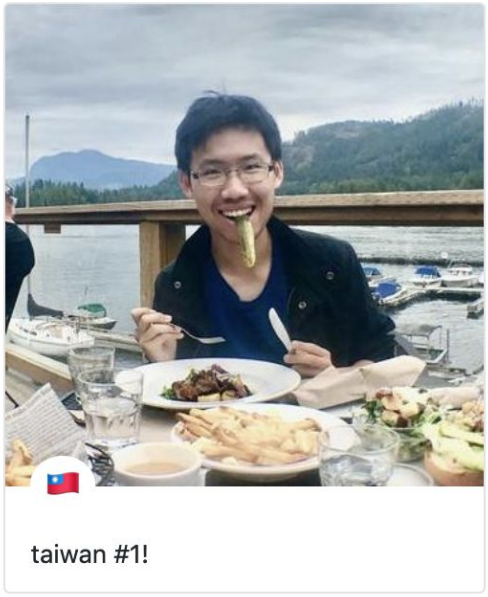
UBC LAUNCH PAD
SOFTWARE ENGINEERING TEAM

UBC CSSS

Slack Workspace:

https://ubccsss.org/projecthub-chat

BRINGING AN IDEA TO LIFE



taiwan #1!

- **Robert Lin** @bobheadxi
- **BSc Mathematics** 3rd year
- **Launch Pad Tech Lead** Web + Moonshot
- Interested in backends, DevOps, and distributed systems

TODAY'S KEY POINTS

1. Why build "personal projects"?
2. Ideation
3. Putting things together
4. Long-term considerations

“YOU SHOULD MAKE PERSONAL PROJECTS!”

...but why?


40 hours of leetcode a week
LeetCode
HackerRank
alamy stock photo
F8ER1E
www.alamy.com


I DEMOED
MLH
Red Bull
Red Bull
Red Bull
30
amazon.com
Gift Card
MLH
XBOX ONE
Microsoft

1. WHY?

PERKS

Beautiful GitHub profile

Things to spice up your resume

“I’ve written some Python”

“I can object oriented blah blah”


Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mon
Wed
Fri
Less
More
Learn how we count contributions.

CONSIDER...

What is software engineering?

The code you write matters

A lot of cool stuff to learn

Problem solving comes with practice

CONVINCE YOURSELF

convince yourself of your goals

2. IDEATION

QUESTIONS TO ASK

What am I interested in?

What sounds cool?*

What sounds fun?

DON'T ASK

What programming language am I going to use?

Omg it sounds hard can I actually do it?

Will it make me money?

KEEP IN MIND

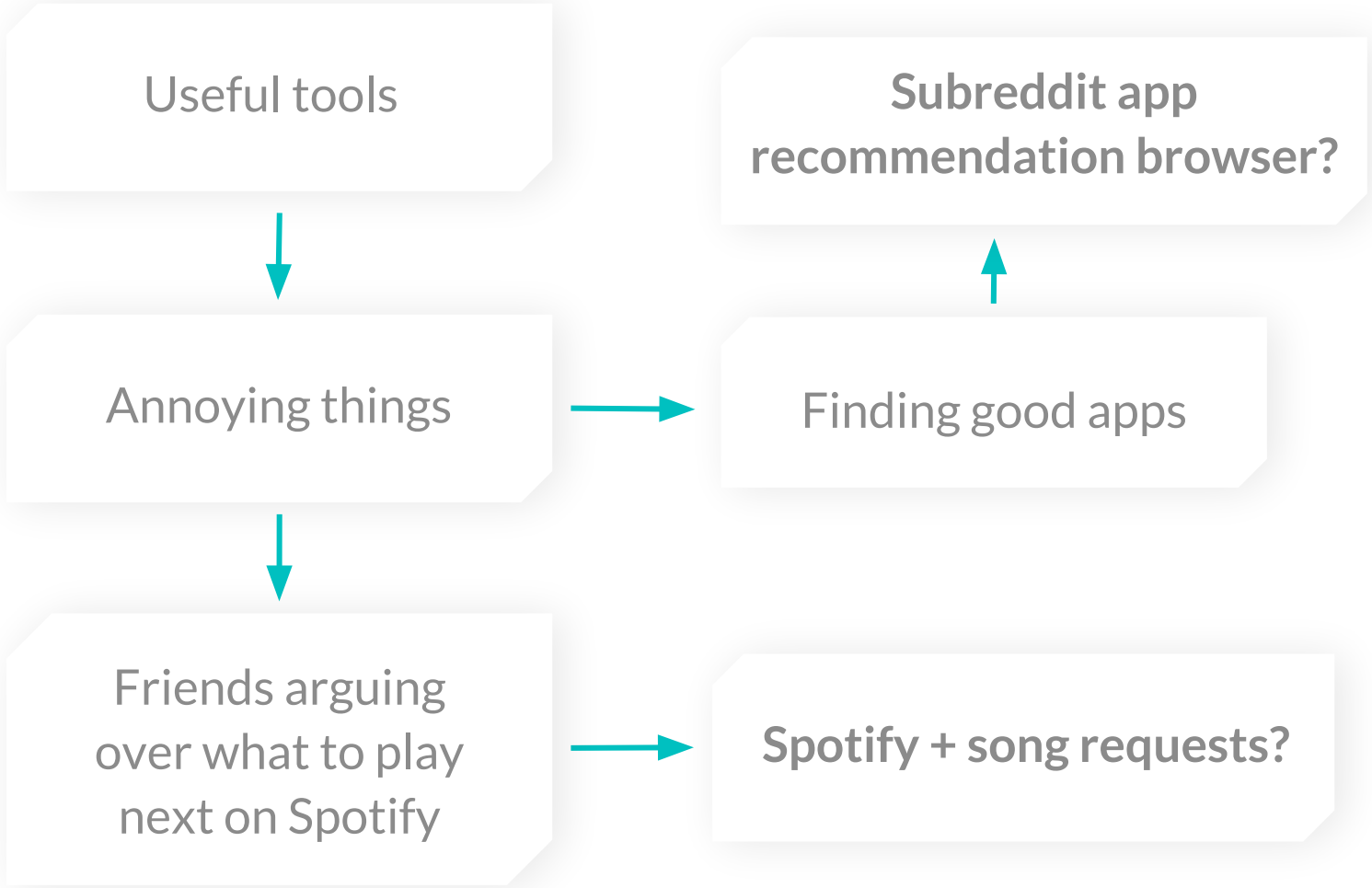
Simple is good

Simple is probably harder than you think

Hard is probably easier than you think*

It doesn’t have to be super unique at first

What are your goals?


Useful tools
Annoying things
Friends arguing over what to play next on Spotify
Finding good apps
Subreddit app recommendation browser?
Spotify + song requests?

BOB'S THOUGHTS

Websites aren't always the best first project

You don't have to do it yourself

It doesn't have to be your project to start with*

Define your features - a "MVP"

“MVP”

Minimum Viable “Product”*

A set of core features / pieces of functionality

Could just be one thing

Less is more

It doesn’t have to be fantastic

Friends arguing over what to play next on Spotify

Spotify + song requests?

- Log in
- Create a playlist
- Let users request songs for a particular playlist
- Allow creator to approve songs

3. PUTTING THINGS TOGETHER

BREAK IT DOWN

Look at each of your features and think about how they might work

Consider your interface

- web/mobile app?
- command line?
- messenger?

RESEARCH

Learn about the pieces you'll need

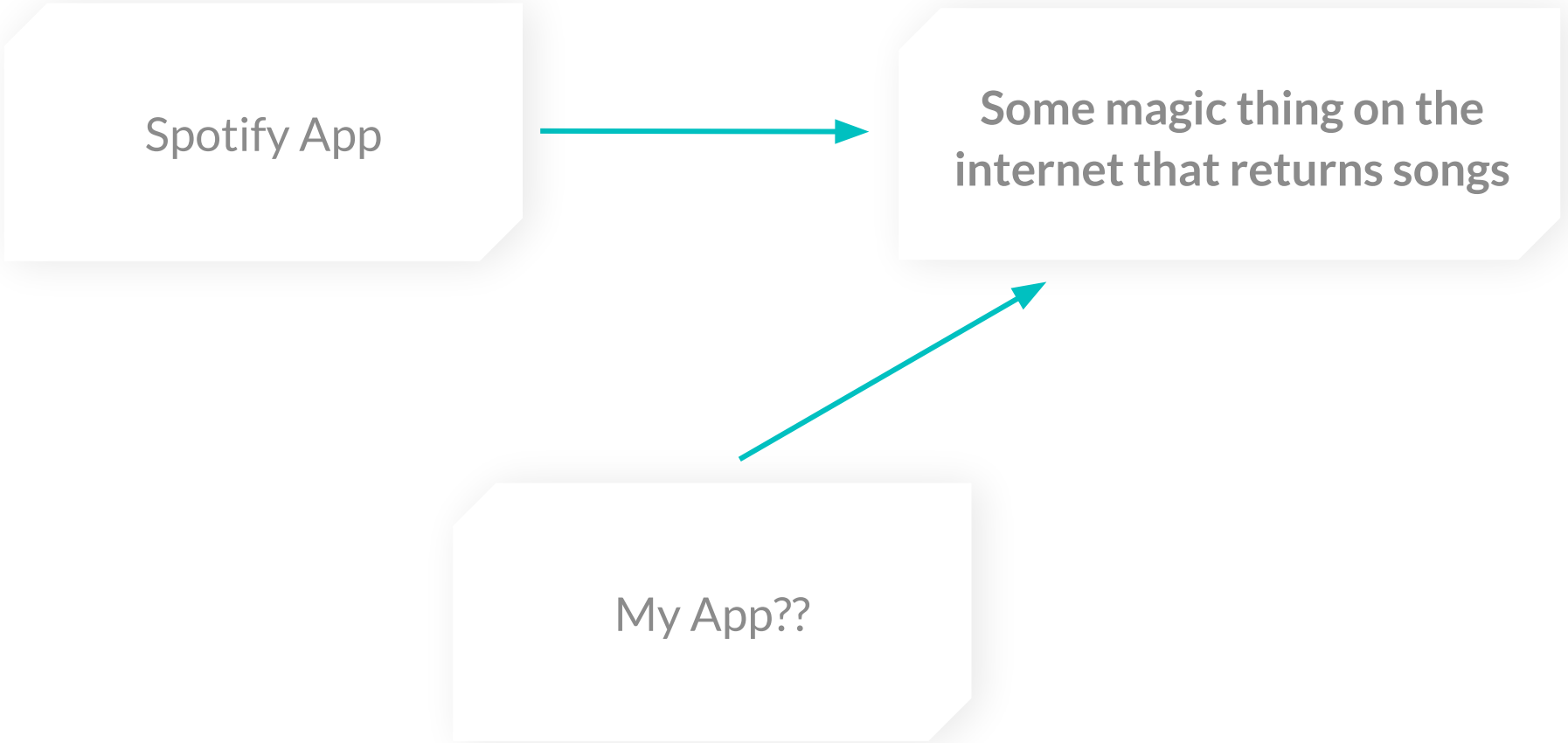
- web/mobile app: how will it interact with things?

Look at examples!

- similar projects on GitHub
- how does Spotify work?

Spotify App

Some magic thing on the internet that returns songs


Spotify App
Some magic thing on the internet that returns songs
My App??

BOB'S THOUGHTS

Be patient

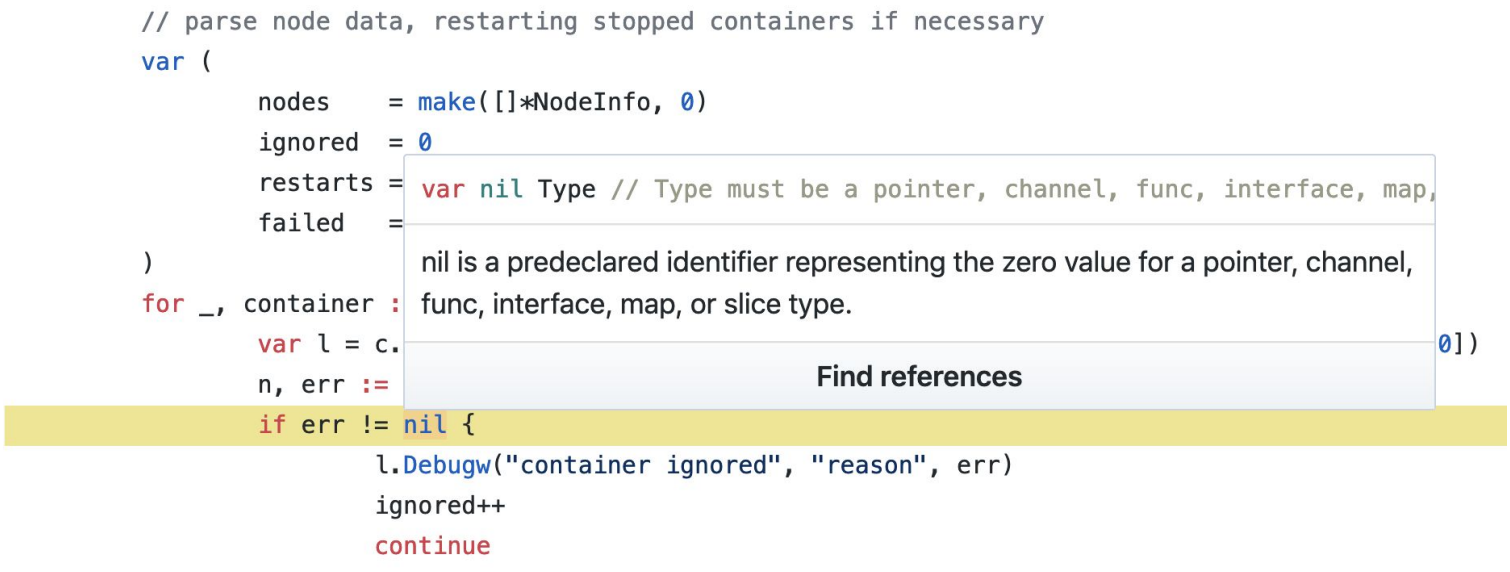
You won't know everything

Don't be afraid of reading source code

Aim for a high-level understanding - try to visualize the pieces of an example

Make use of on-hover tooltips and the ability to jump into the source for a function (cmd-click or something)

- Browser: Sourcegraph extension
- Code Editor: Visual Studio Code, IntelliJ, etc.

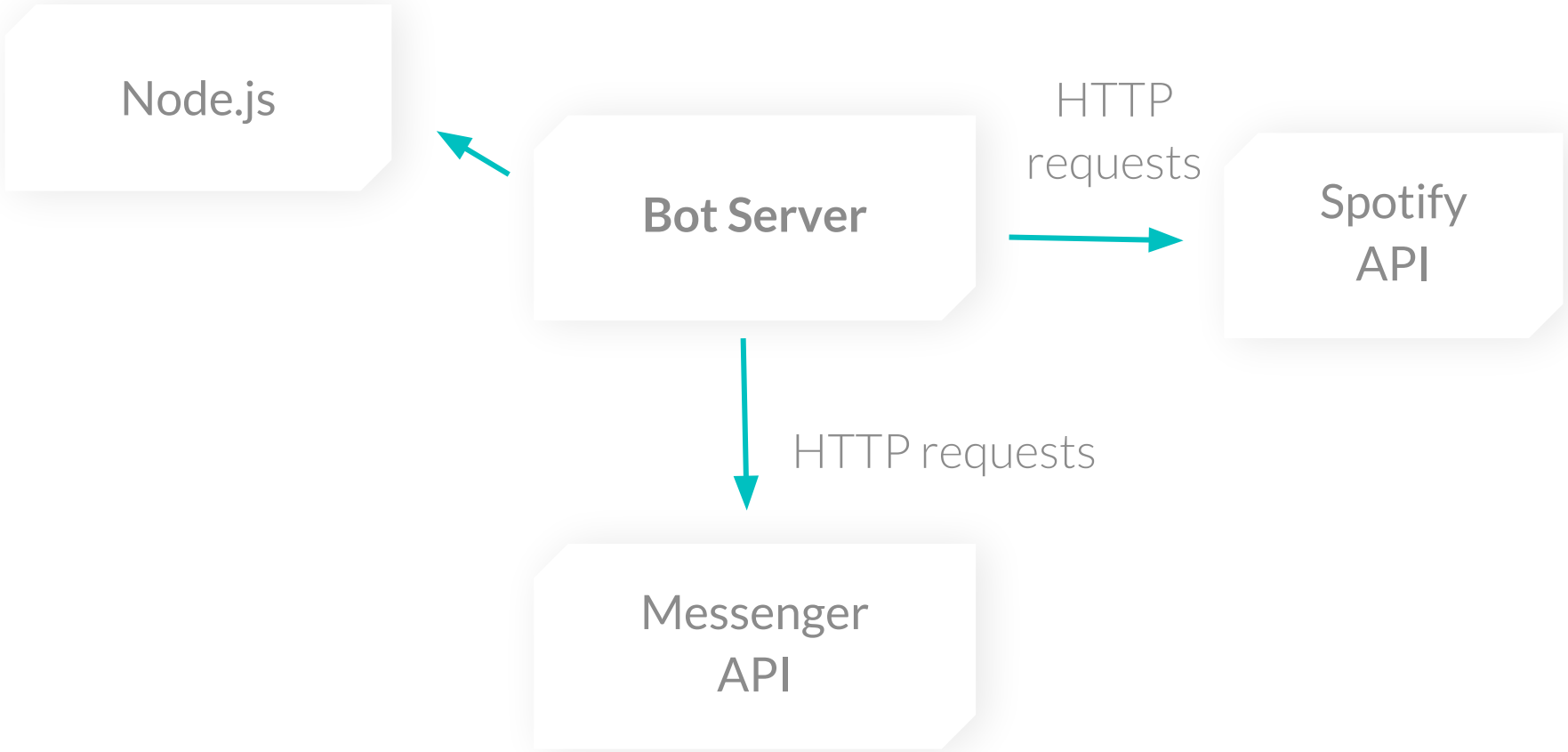


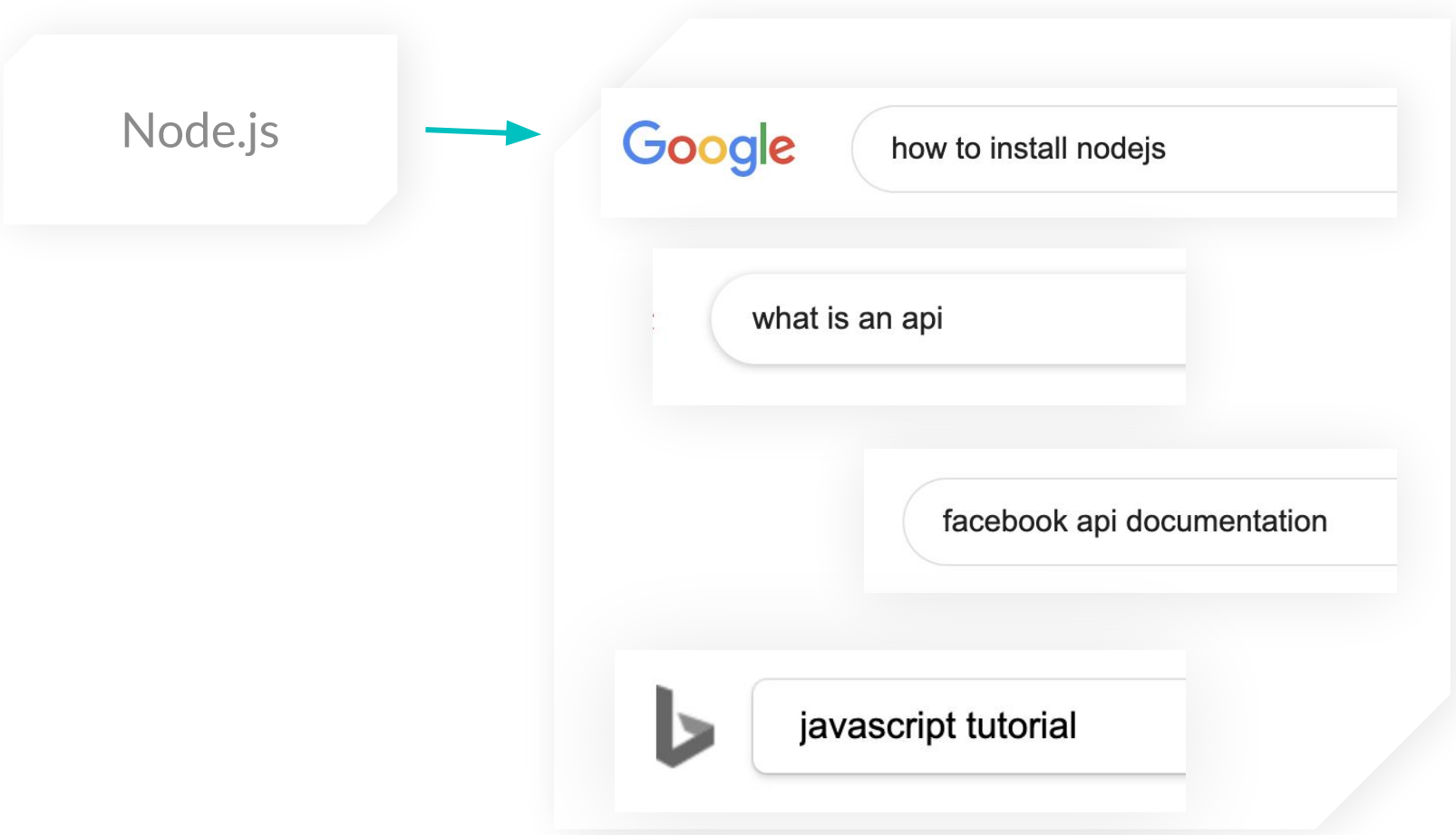
BOB'S THOUGHTS

Don't be too opinionated about tech

Don't worry about people saying "oh Python/MongoDB/x is slow"

The goal is to put something together


Node.js
Bot Server
HTTP
requests
Spotify
API
HTTP requests
Messenger
API


Node.js
Google
how to install nodejs
what is an api
facebook api documentation
javascript tutorial

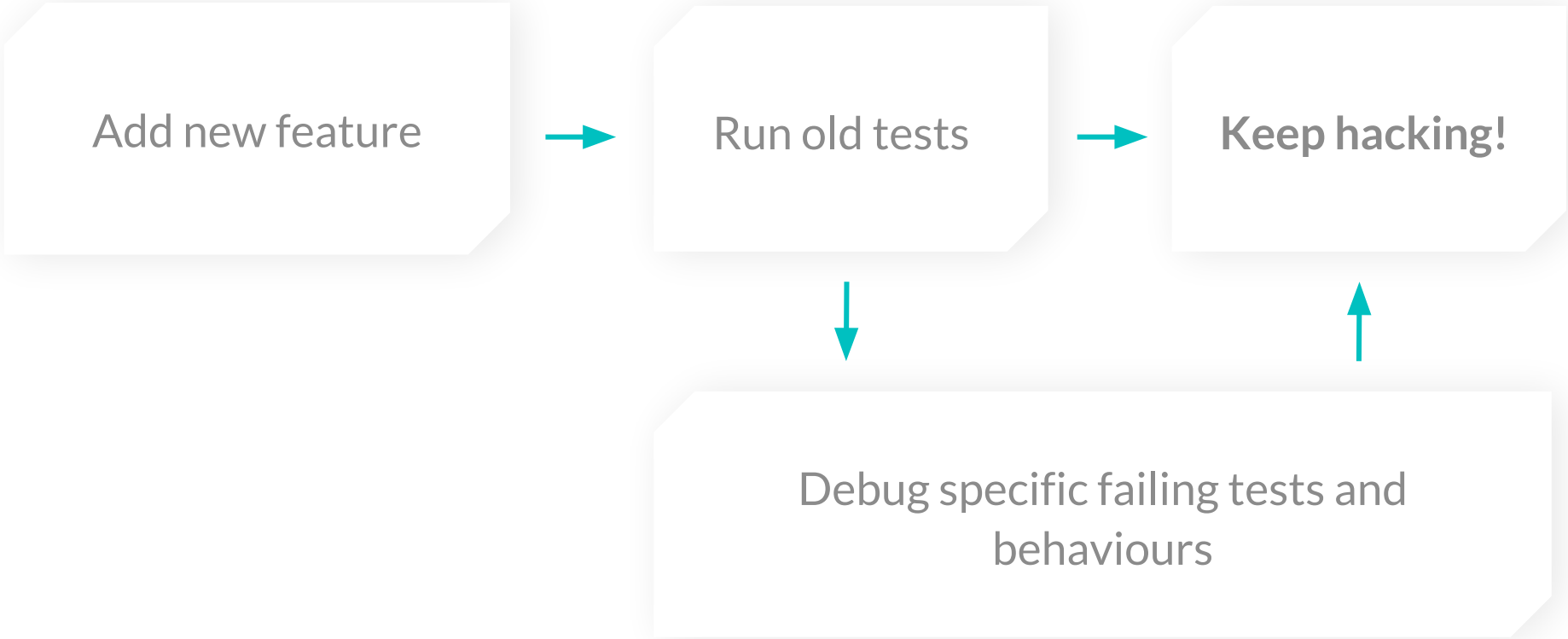
4. LONG-TERM CONSIDERATIONS

WHAT IS CODE?

Instructions for the computer, and you!

You want to be able to add new features without destroying everything

You want other people to be able to look at it (maybe)


Add new feature
Run old tests
Keep hacking!
Debug specific failing tests and behaviours

DEPLOYING

Set up a clear, automated procedure

https://github.com/ubclaunchpad/inertia

(or some scripts, Heroku, etc….)

Programming means getting a program working. You have a problem to solve, **you write some Go code, you run it, you get your answer, you're done. That's programming, and that's difficult enough by itself. But what if that code has to keep working, day after day?** What if five other programmers need to work on the code too? Then you start to think about version control systems, to track how the code changes over time and to coordinate with the other programmers. You add unit tests, to make sure bugs you fix are not reintroduced over time, not by you six months from now, and not by that new team member who's unfamiliar with the code. You think about modularity and design patterns, to divide the program into parts that team members can work on mostly independently. You use tools to help you find bugs earlier. You look for ways to make programs as clear as possible, so that bugs are less likely. You make sure that small changes can be tested quickly, even in large programs. You're doing all of this because your programming has turned into **software engineering**. - Russ Cox

EXTRA

THANKS!