1. Talk about Identity & Auth
2. Hash Tables
3. Peer Review
   Split back into project teams
4. PPH
5. Lunch
6. 2pm: CCW #4 W/Robin
   Behavioral Interviewing

Hash Tables

Basically (kinda/sorta) Key Value Storage

.set(“John”, 52)
.set(“Cathy”, 50);
.set(“Zach”, “Linfield”);

.has(“John”); <= true
.has(“Allie”); <= false
// .contains()
// .includes()

.get(“John”); <= 52

Let family = {
  John: 52,
  Cathy: 50,
  Zach: “Linfield”
}

family.john

family.get(“John”);

(1) => (4) => (5) => (1) => (7)

.set(1, 1)
.has(4)?
.has(5)?
.has(1)? T
  .set(1, 2)
.has(7)

Efficiency:

O(1)

Buckets

HashTable myFamily = new HashTable(2048)

[

0 - () -> null

1 -

2 - Cathy: 50, Miriam:NaN, Ed:33

3 -

4 - Zachary:Linfield

]

myFamilly.set(“John”, 52);

Key to this is a “hashing algorithm”
- Turn the key into a number
  - Take every letter and get ascii#
  - Add that all up
  - Multiply by a prime number
  - Divide by # of buckets

.set(key, val)
1: Hash the key
2: Do we have something there?
  - Either: Make a list ...
  - Or … rebuild the whole thing
2: Store the key/value in the ###

.has(key)
1: Hash the key
2: Is there a list at that key?
3: Go through the list and find…