Exploding Chickens

A full-stack card game

Radison Akerman, November 2022

Hi there, my name is

Radison Akerman.

Project Manager & Info Sec @ UIC CoE
CS + Business @ UIC

EXPLODING CHICKEN

SHOW THIS CARD
IMMEDIATELY.

YOLKING AROUND

Draw Deck x52   Discard Deck x0

Exploding Chicken

Avoid this little guy at all costs. Once this card is drawn, you must use a defuse card to stop the ticking time bomb. If you don't have a defuse card, it looks like your time is up.

Exploding Chicken

```
if (card is drawn)
  if (player has defuse)
    plays defuse, places
    chicken back in draw deck
else
    player explodes, removed
    from game permanently
```

Defuse Card

```
if (card is drawn)
    place card in players hand

if (card is played &&
player is exploding)
    discard card
    && prompt player to place
    chicken back in draw deck
    && advance turn
```

DEFUSE
VIA CHICKEN DANCE

FIRST EDITION

DEFUSE
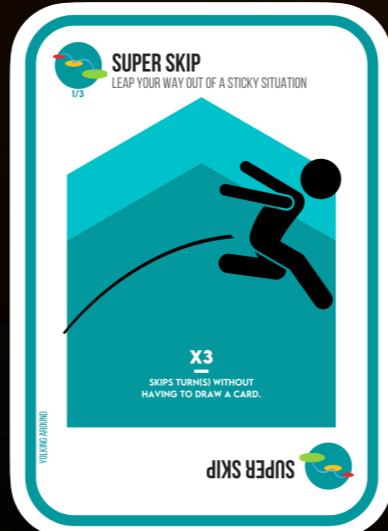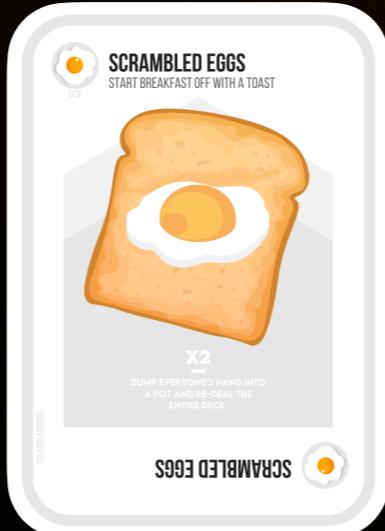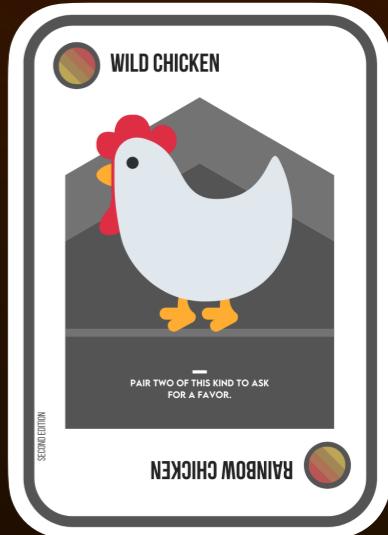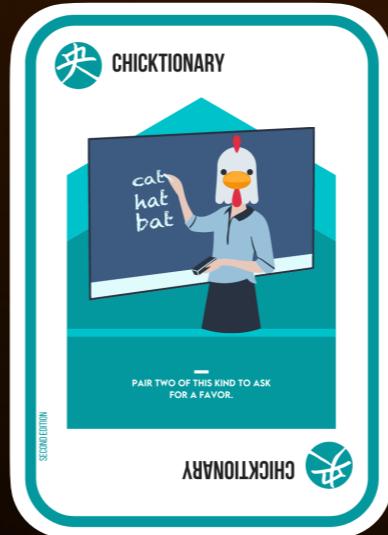
Shuffle Card

```
if (card is drawn)
    place card in players hand

if (card is played)
    shuffle draw deck
    && discard card
```

Shuffle Card

- Player wants to play card
- Sends "play-card" request
- Cascading validation phase
  - Is the player valid?
  - Is it their turn?
  - Do they have this card?
  - Can they play it now?
- Completes card action (randomize draw deck)
- Discard card
- Tell everyone what happened

EXPLODING CHICKEN
SHOW THIS CARD IMMEDIATELY.
EXPLODING CHICKEN

DEFUSE
DUCK... DUCK... DEFUSE!
X6
STOP AN EXPLODING CHICKEN AND PUT IT BACK IN THE DECK.
DEFUSE

ATTACK
YEET A CHAIR
X4
FORCE THE NEXT PLAYER TO TAKE TWO TURNS.
ATTACK

SEE THE FUTURE
YOU'RE A WIZARD, HARRY!
X5
VIEW THE NEXT THREE CARDS IN THE DRAW PILE.
SEE THE FUTURE

SKIP
CUT THE LINE AT THE BANK
X4
SKIP YOUR TURN TO THE NEXT PLAYER.
SKIP

REVERSE
LAND THE BACKFLIP 180
X4
FLIP THE TURN ORDER AND END YOUR TURN.
REVERSE

SHUFFLE
PRODUCE A FIRE EDM MIX
X4
TAKE THE DECK AND SHUFFLE IT. PUT THE DECK BACK.
SHUFFLE

FAVOR
ASK YOUR PROFESSOR WHAT'S ON THE EXAM
X4
ASK A PLAYER FOR A FAVOR IN THE FORM OF A CARD.
FAVOR

RAINBOW CHICKEN
PAIR TWO OF THIS KIND TO ASK FOR A FAVOR.
RAINBOW CHICKEN

CHICKTIONARY
PAIR TWO OF THIS KIND TO ASK FOR A FAVOR.
CHICKTIONARY

SURFING CHICKEN
PAIR TWO OF THIS KIND TO ASK FOR A FAVOR.
SURFING CHICKEN

WILD CHICKEN
PAIR TWO OF THIS KIND TO ASK FOR A FAVOR.
RAINBOW CHICKEN

HOT POTATO
DROP A STEAMING HOT POTATO ON A GOOD FRIEND
1/1
X1
PASSES THE TURN ON TO THE NEXT PLAYER. ONLY PLAYABLE WHEN AN EC IS DRAWN.
HOT POTATO

FAVOR GATOR
BECOME LOW-KEY
1/2
X2
STOP THE FAVOR OF ANOTHER PLAYER. THE PLAYER MUST GIVE YOU ONE CARD.
FAVOR GATOR

SCRAMBLED EGGS
START BREAKFAST OFF WITH A TOAST
1/2
X2
DUMP EVERYONE'S HAND INTO A POT AND RE-DEAL THE ENTIRE DECK.
SCRAMBLED EGGS

SUPER SKIP
LEAP YOUR WAY OUT OF A STICKY SITUATION
1/3
X3
SKIPS TURN(S) WITHOUT HAVING TO DRAW A CARD.
SUPER SKIP

SAFETY DRAW
SWERVE AROUND THE TRAFFIC CONE
1/4
X4
ENDS YOUR TURN BY DRAWING THE FIRST CARD THAT IS NOT AN EXPLODING CHICKEN.
SAFETY DRAW

DRAW FROM THE BOTTOM
PULL OUT ONE OF THE LARGE BOXES
1/4
X4
END YOUR TURN BY DRAWING A CARD FROM THE BOTTOM OF THE DECK.
DRAW FROM THE BOTTOM

900 Games
7,000 Minutes
31,000 Cards Played

Since April 2021

Demo

Type in your web browser...
rakerman.com/link/demo

Type in your web browser...
rakerman.com/link/backup

Scalable and fast

Room for expansion

Easy to understand

Where do you start?

Maximize compatibility
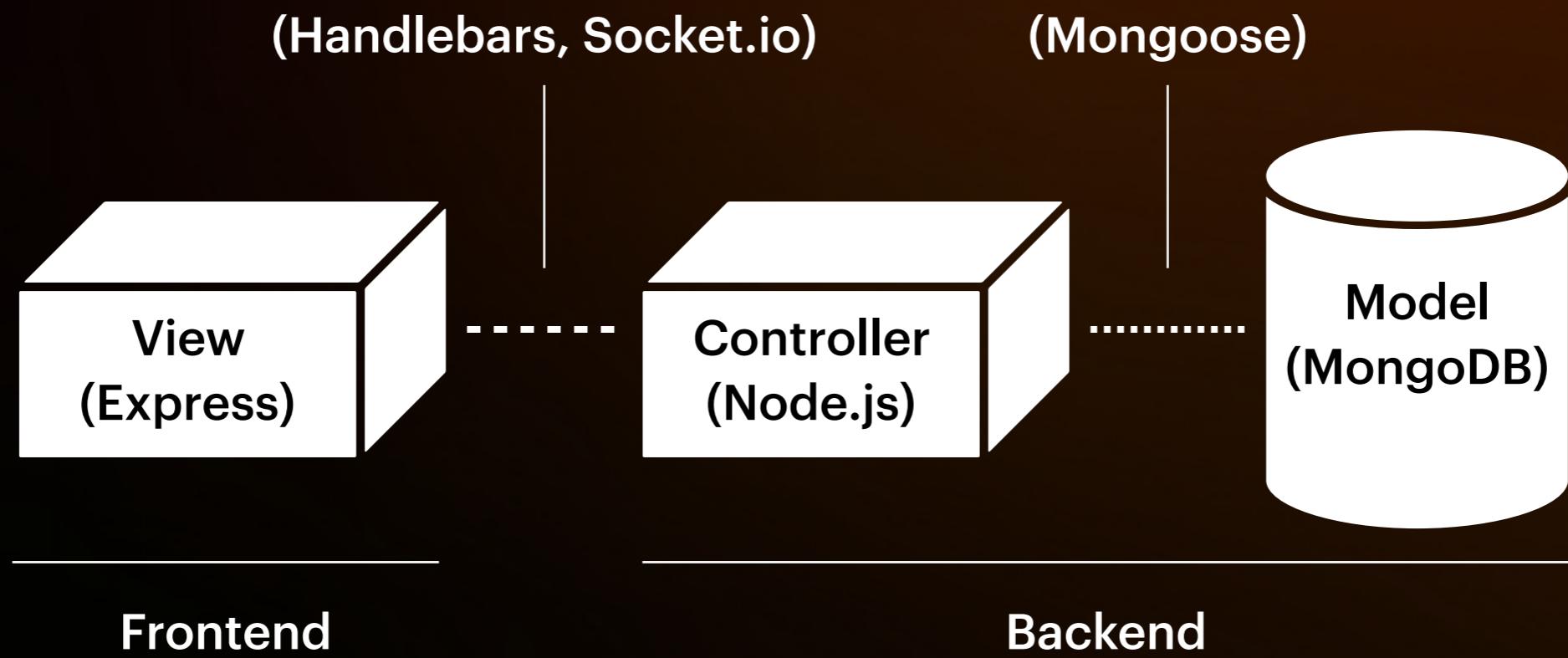
Quick to develop

Real-time rendering
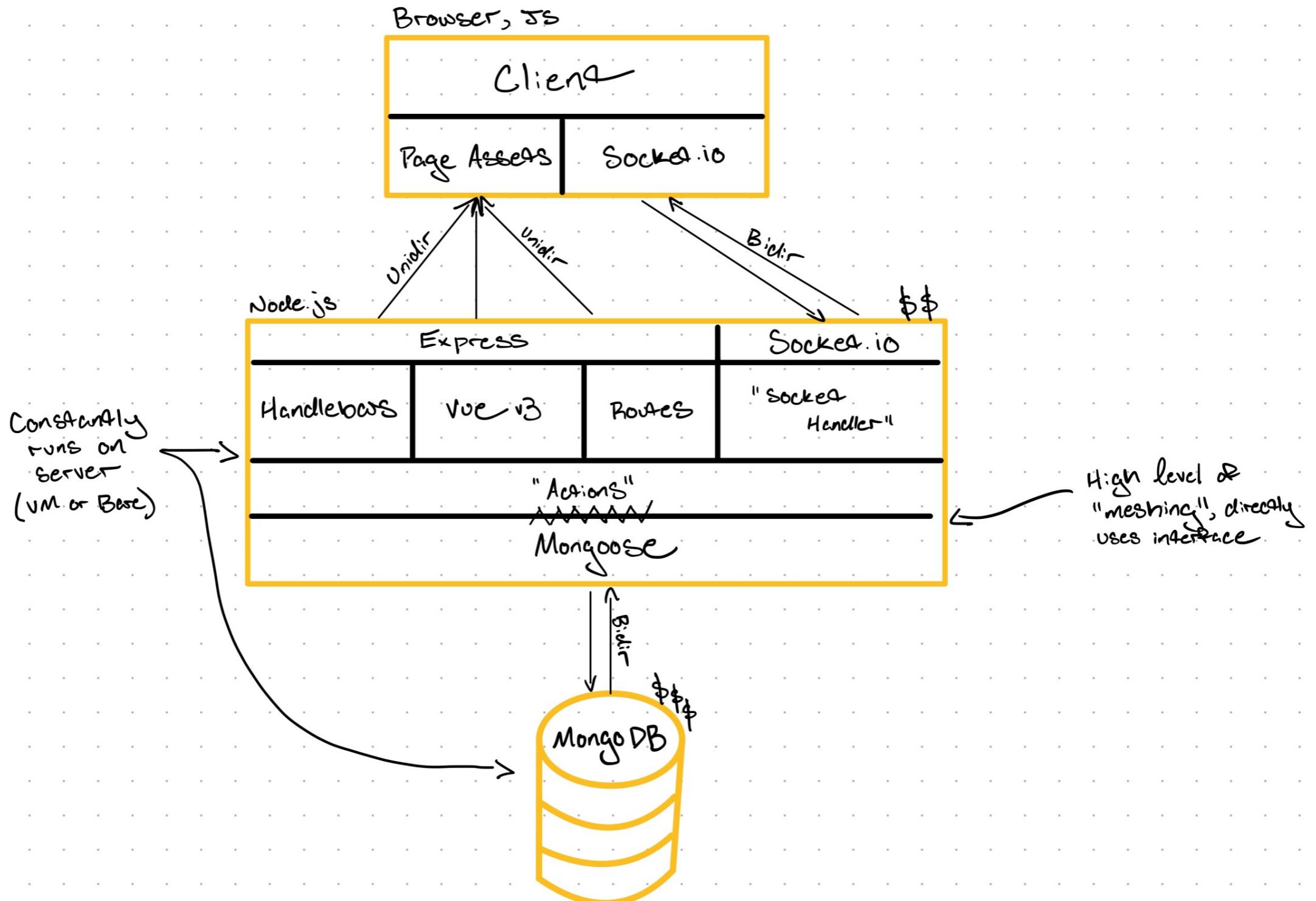
v1.0.0
An unorganized first attempt (that works)

- Full-stack web application

  - Node.js — an asynchronous event-driven JavaScript runtime geared towards scalable network apps

  - MongoDB — a NoSQL document-oriented database

    - Mongoose as an ODM (Object Data Modeling)

  - Handlebars — HTML templating language

  - Socket.io — real-time, bidirectional communication

(Handlebars, Socket.io)   (Mongoose)

View
(Express)   Controller
(Node.js)   Model
(MongoDB)

Frontend   Backend

Exploding Chickens

Browser, JS

Client

| Page Assets | Socket.io |

Unidir → Unidir → Bidir

Node.js

$$

Express | **Socket.io**

| Handlebars | Vue v3 | Routes | "Socket Handler" |

Constantly runs on server (VM or Bare)

"Actions"

Mongoose

High level of "meshing", directly uses interface

Bidir

Mongo DB $$$

Game

Cards

- _id
- action
- assignment
- position
- pack

Players

- _id
- nickname
- avatar
- seat_position
- wins
- sockets_open
- is_host
- is_dead

Events

- _id
- tag
- req_player
- target_plyr
- related_key
- related_value

Misc data...

Socket.io Client

↓ ↑

socket-handler.js

game-actions.js

card-actions.js

Shuffle Card

- Player wants to play card
- Sends "play-card" request
- Cascading validation phase
  - Is the player valid?
  - Is it their turn?
  - Do they have this card?
  - Can they play it now?
- Completes card action (randomize draw deck)
- Discard card
- Tell everyone what happened

v2.0.0
A segmented second attempt (much, much cleaner)

- Same web stack (Node.js, MongoDB, Handlebars, Socket.io) + Vue.js + Auth0

- Mocha + Istanbul — Backend tests, code coverage

- Shared repository model

- Issues and features request tracking in Github

- New Lobby abstraction

Lobby

Games

Cards

Events

Players

- _id
- **game_assign**
- nickname
- avatar
- seat_position
- wins
- sockets_open
- is_host
- is_dead

Events

- _id
- tag
- req_player
- target_plyr
- related_key
- related_value

Misc data...
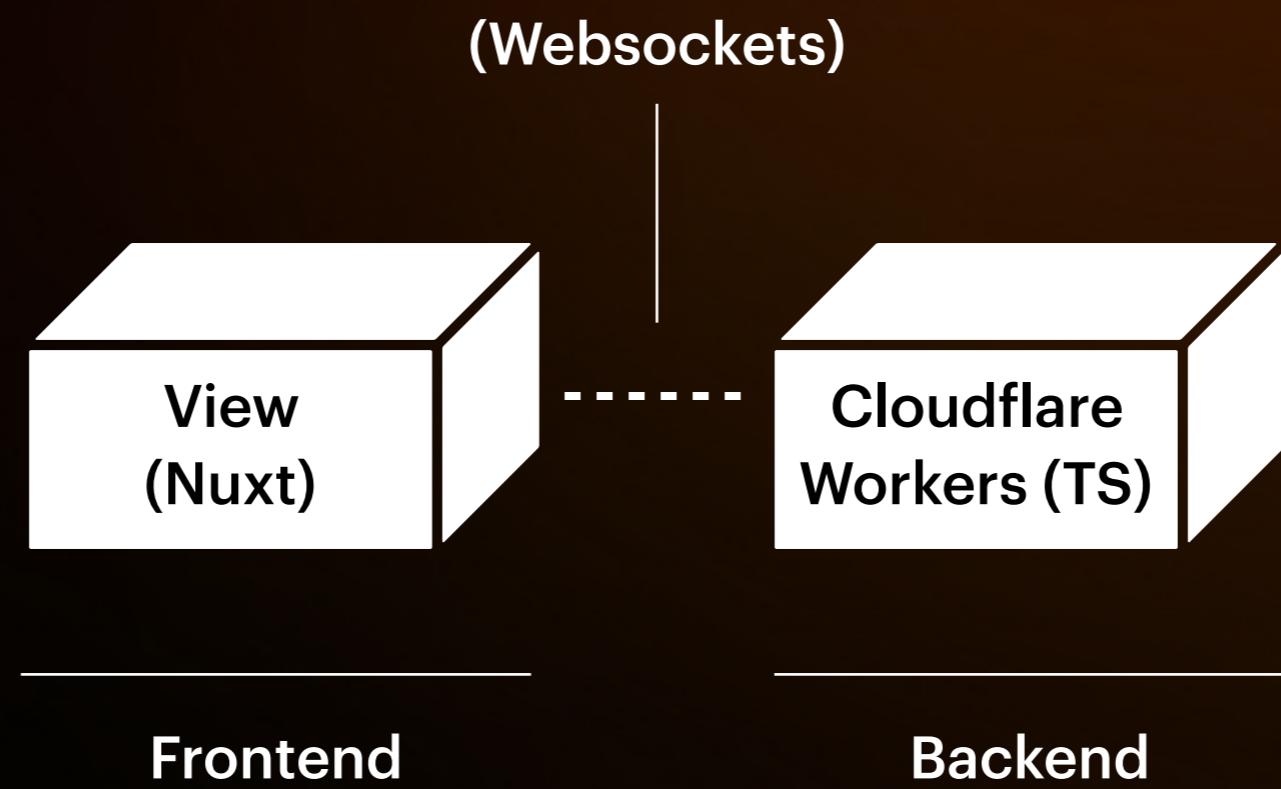
Releases
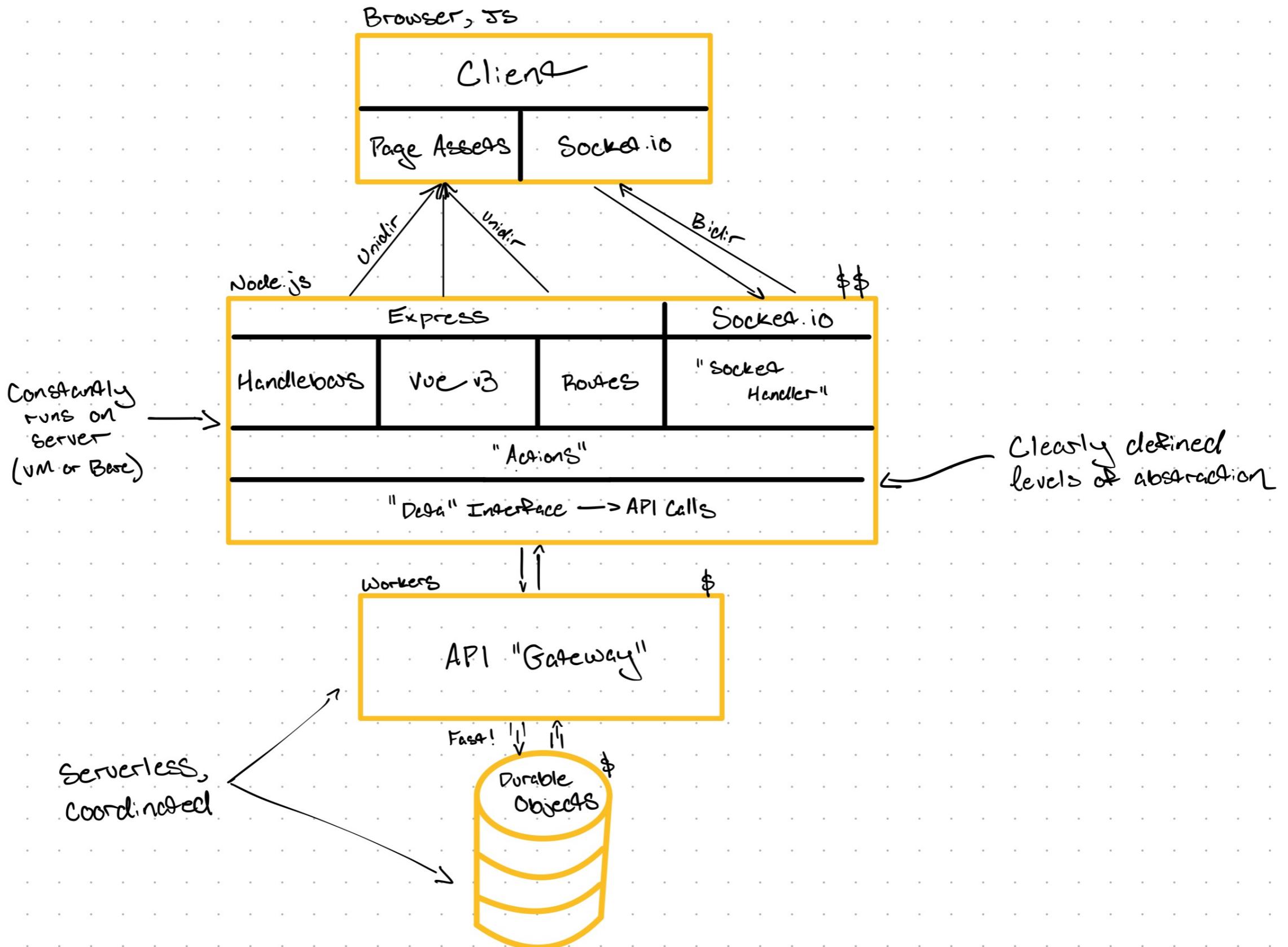
v3.0.0
A globally-distributed 3rd rework (here we go again)

- Completely new web stack: Typescript (OOP) + Cloudflare Workers + Websockets + Nuxt + Auth0

  - State-of-the-art, lightweight & serverless = cheap

  - Ability to run 'on the edge', decentralized (worldwide)

- Same test coverage, lobby abstraction, and user interface

Exploding Chickens   Stack v2.1

Browser, JS

Client

| Page Assets | Socket.io |

Unidir — Unidir — Bidir $$

Node.js

Express			Socket.io
Handlebars	Vue v3	Routes	"Socket Handler"

"Actions"

"Data" Interface ——> API Calls

Constantly runs on server (VM or Bare)

Clearly defined levels of abstraction

Workers $

API "Gateway"

Fast!

Durable Objects $

Serverless, coordinated

Exploding Chickens   Stack v3.0

Browser, JS

Client

| Page Assets | Socket.io |

↑ Unidir

↘↖ Bidir $

Workers

| Nuxt v3 | Socket.io |
| | "Socket Handler" |

"Actions"

Data "Gateway"

Serverless,
coordinated →

Fast! ↓↑ $

Durable
Objects

Key Takeaways

- Do your research when picking a language

- Build a solid data structure

- Clearly define boundaries between parts of your project

- Give yourself room to build something better

- Visualize everything upfront

chickens.rakerman.com