Date _____ 20 _____

Microservices

Monolith

all routing, middleware, biz logic & DB access to implement all features of app

A single Microservice contains...

all routing middleware, biz logic & DB access to implement one feature of app

Data management b/w services big problem

↓

the way we store data in service & how to communicate b/w services

w/ microservices we store & access data in a strange way

Date _____ 20 ___

each service gets its own DB (if needed)

services will never reach into another services DB

Database-per-Service

- we want each service to run independently
- DB schema/structure might change unexpectedly
- some services might function more efficiently w/ diff type of DB (SQL v noSQL)

Communication Strategies b/w services

Sync → services communicate w/ each other using direct requests

Async → services communicate w/ each other using events

Date _____ 20 ____

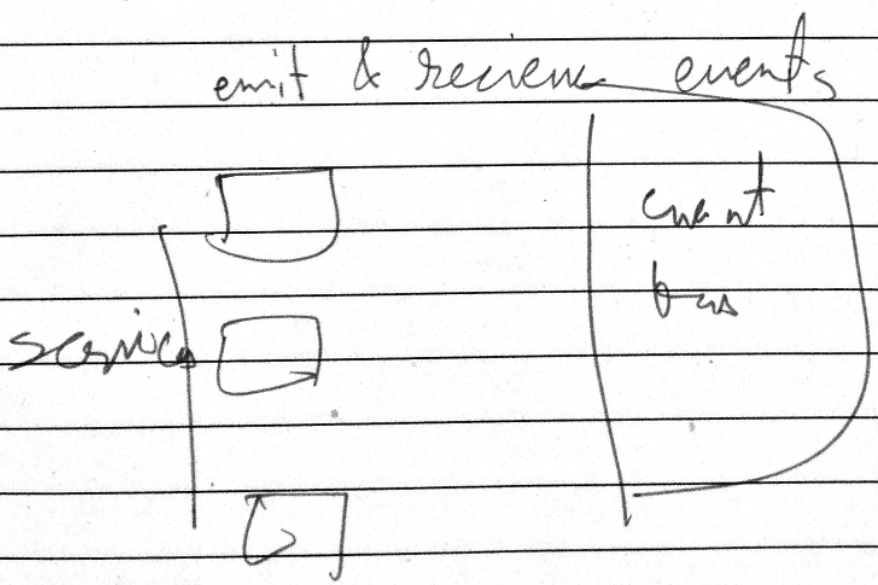
Sync communication

- easy to understand
- might not need DB
- introduces dependencies b/w services
- if any inter service request fails overall request fails
- entire request only fast as slowest request
- can introduce web of requests

Async communication

event Bus

emit & receive events



Sharjeel

Date 20

event bus & DB

emit event when storing to DB.

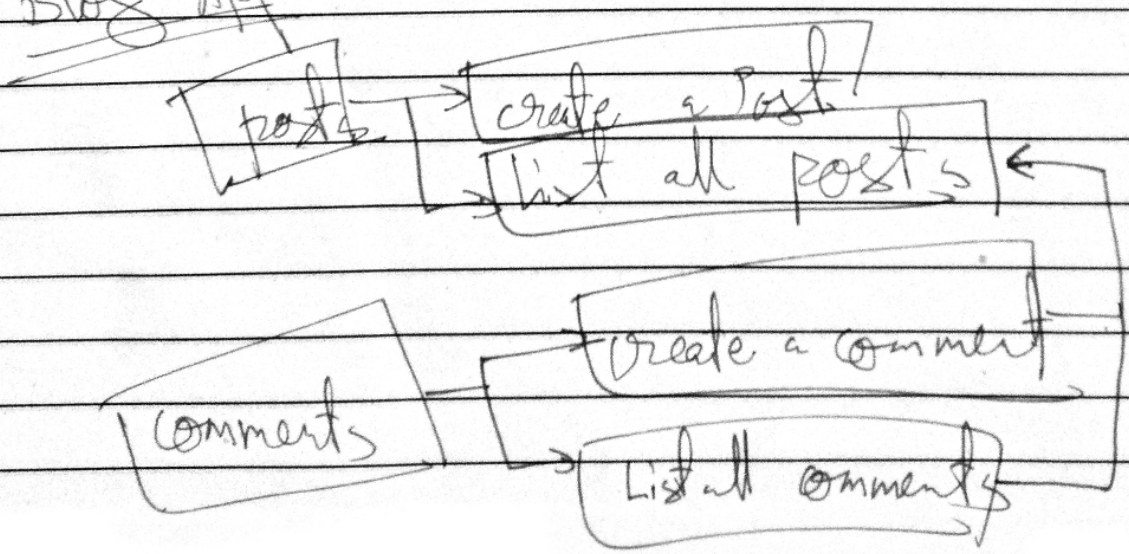
event bus emits to any interested services that have subscribed

subscribed service takes event and adds info to DB

Cons

data duplication extra storage extra DB

harder to understand



Date _____ 20 ____

Event Bus

RabbitMQ, Kafka, NATS

recieve events, publishes them to listeners

Date _____ 20 ____

Dealing w/ Missing Events

Service down - did not recieve events

opt 1 sync request

need code in each service

opt 2 direct DB access

need code in down service

opt 3 store events

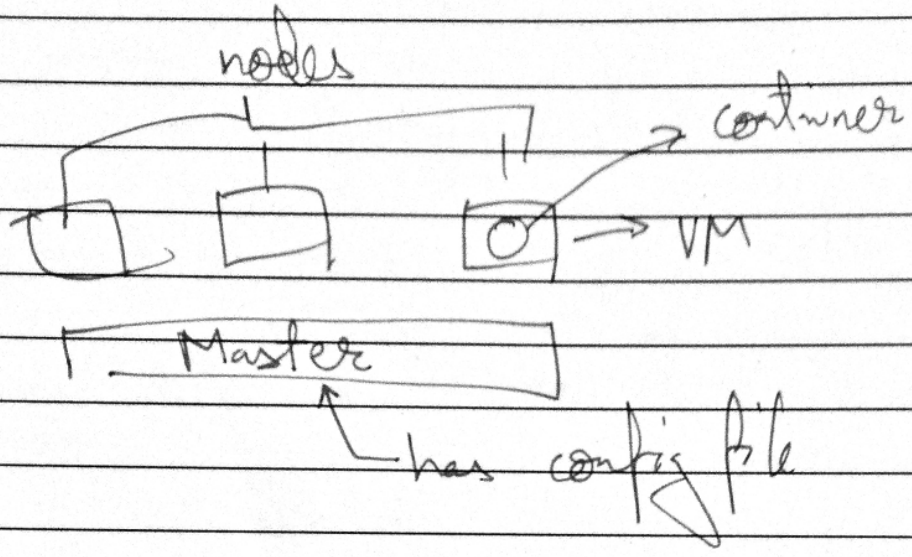
Event bus data store

Event Sync

Date _____ 20 ____

Kubernetes run bunch of different containers

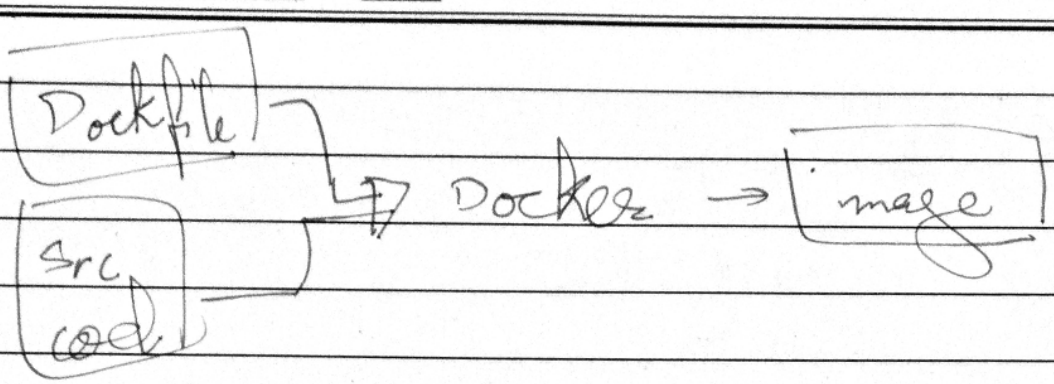
Kubernetes Cluster



Kubernetes forwards requests to service

launching new instances & scaling services made easier

Date _____ 20 ____



Kubernetes manages networking

pods wrap containers)

service
↳ handles networking

Date _____ 20 ____

Kubernetes Cluster — collection of nodes & a master to manage them

Node — A virtual machine that will run our containers

Pod — A wrapper over a running container. can house multiple containers.

Deployment — Monitors a set of pods makes sure they are running & restarts them if they crash

Service — provides an easy to remember URL to access a running container.

Date _____ 20 ____

Kubernetes Config Files

- Tell K8s about the different Deployments, Pods & Services (Objects)
- written in YAML
- stored w/ src control
- config files provide a precise definition of what your cluster is running

Date K8s config file

```
apiVersion : v1

kind: Pod

metadata :
|       name :

spec :
|   containers :
|   |   - name :
|   |   |   image :
|   |   |   imagePullPolicy :
```

$ kubectl apply -f config.yaml // create pod

$ kubectl get pods

Date _____ 20 ____

docker ps ⟶ kubectl get pods

docker exec -it ⟶ kubectl exec -it

docker logs ⟶ kubectl logs

kubectl apply -f

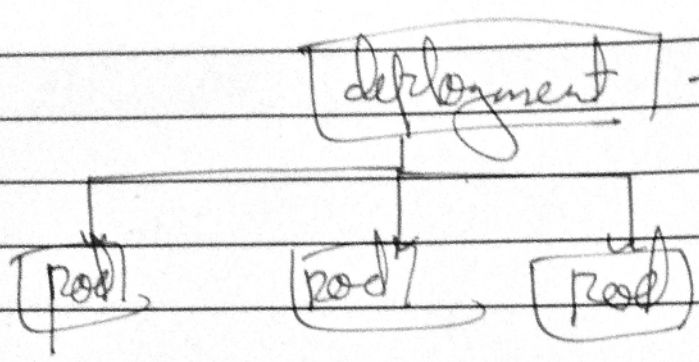
kubectl delete pod

kubectl describe pod

↓

debug pb

Date _____ 20 ____



→ also does version management

depl.yaml

```
apiVersion: apps/v1
Kind: Deployment
metadata:
  | name:

spec:
  | replicas:
  | selector:
  |   | matchLabels:
  |   |   | app:
  | template:
  |   | metadata:
  |   |   | labels:
  |   |   |   | app:
  |   | spec:
  |   |   | containers:
  |   |   |   | name:
  |   |   |   | image:
```

Shafjeel

Date _____ 20 ____

kubectl get deployments

kubectl describe deployment

kubectl delete deployment

↳ also deletes associated pods

updating the image used by deployment

- use latest tag in pod spec section
- make update to src code
- build image
- push to docker hub
- kubectl rollout restart deployment

Shahjeel

Date _____ 20 ____

Services provide networking b/w pods & from outside to pod

Types of Services

Cluster IP
set up easy to remember URL to access a pod. only exposes pods in the cluster

Node Port
Makes a pod accessible from outside the cluster. Usually only used for dev purposes

LoadBalancer
makes a pod accessible from outside the cluster the right way.

External Name
Redirects an in cluster request to a CNAME URL

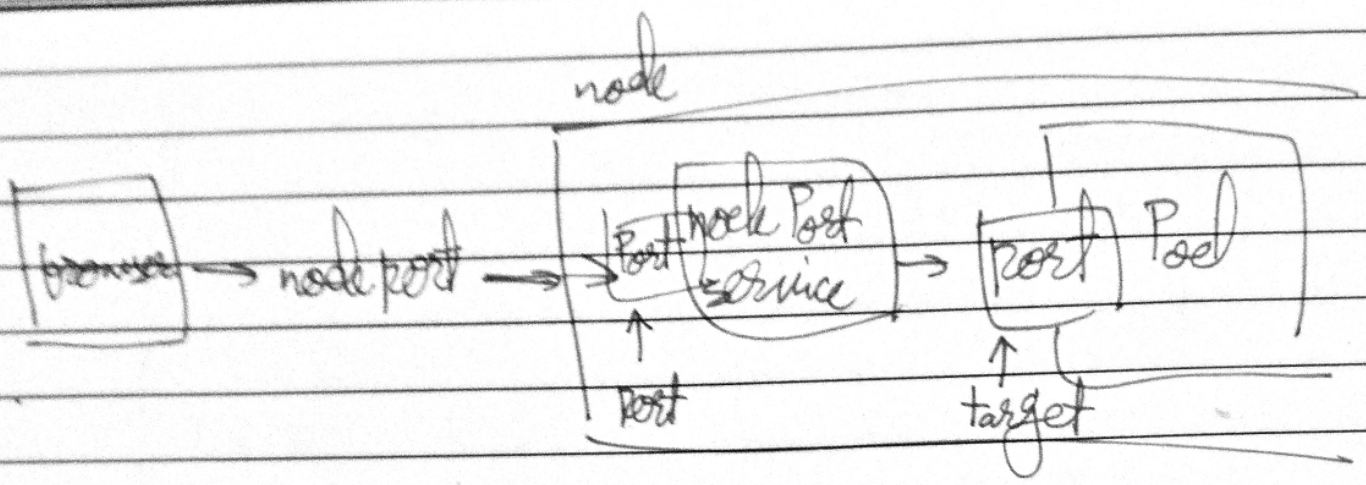
Sharjeel

Date _____ 20 ____

Srv.yaml

```
apiVersion : v1
Kind : Service
metadata :
  name :
spec :
  type : NodePort
  selector :
    app :
  ports :
    - name :
      protocol :
      port :
      targetPort :
```

Date ____ 20 ____



<minikube ip> : 3xxxx

req to http://service-name:Port

Date _____ 20 ____

React app use Load balancer service

↓

point of entry for all services

Load Balancer → tells kubernetes to reach out to its provider & provision a load balancer sets traffic in to a single pod

Ingress A pod w/ a set of routing rules to distribute traffic to other services

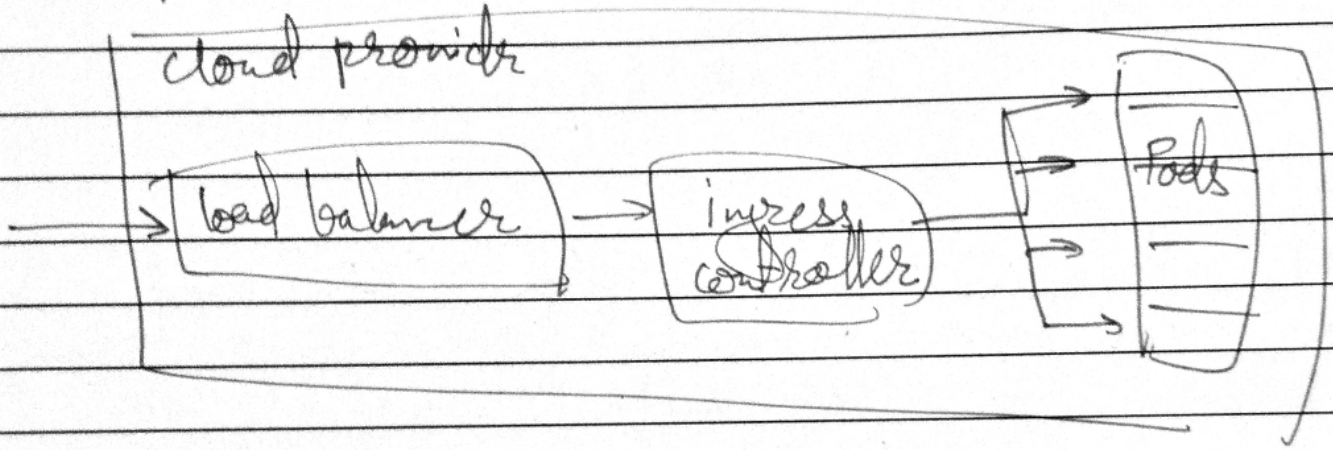
or

Ingress Controller

not exactly the same

Date _____ 20 ____

load balancer ~~exist~~ service asks cloud provider to provision load balancer



Date _____ 20 ____

ingress-srv.yaml

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress

metadata:
  name:
  annotations:
    kubernetes.io/ingress.class: nginx
spec:
  rules:
    - host:
      http:
        paths:
          - path:
            backend:
              serviceName:
              servicePort:
```

Date _____ 20 ____

Skaffold

runs outside cluster

- Automates many tasks in a Kubernetes dev environment
- makes it really easy update code in a running pod
- makes it easy to create/delete all objects tied to a project at once

skaffold.yaml

```
apiVersion : skaffold/v2alpha3
kind : Config
deploy :
    kubectl :
        manifests :
            - <path to K8s config files>
build :
    local :
        push : false
    artifacts
        - image :
            context
            docker :
                dockerfile
            sync :
                manual
                    - src :
                      dest :
```

Date _____ 20 ____

~ skaffold lec

lessons learned

- the fix challenge in microservices is data
- different ways to share data between services
  focus on async
- Async communication focuses on communicating changes using events sent to an event bus
- Async communication encourages each service to be 100% self sufficient. relatively easy to handle temporary downtime or new service creation
- docker makes it easier to package up services
- Kubernetes is a pain to set up but makes it easy to deploy + scale services

Shahjeel

Date _____ 20 ____

cons so far

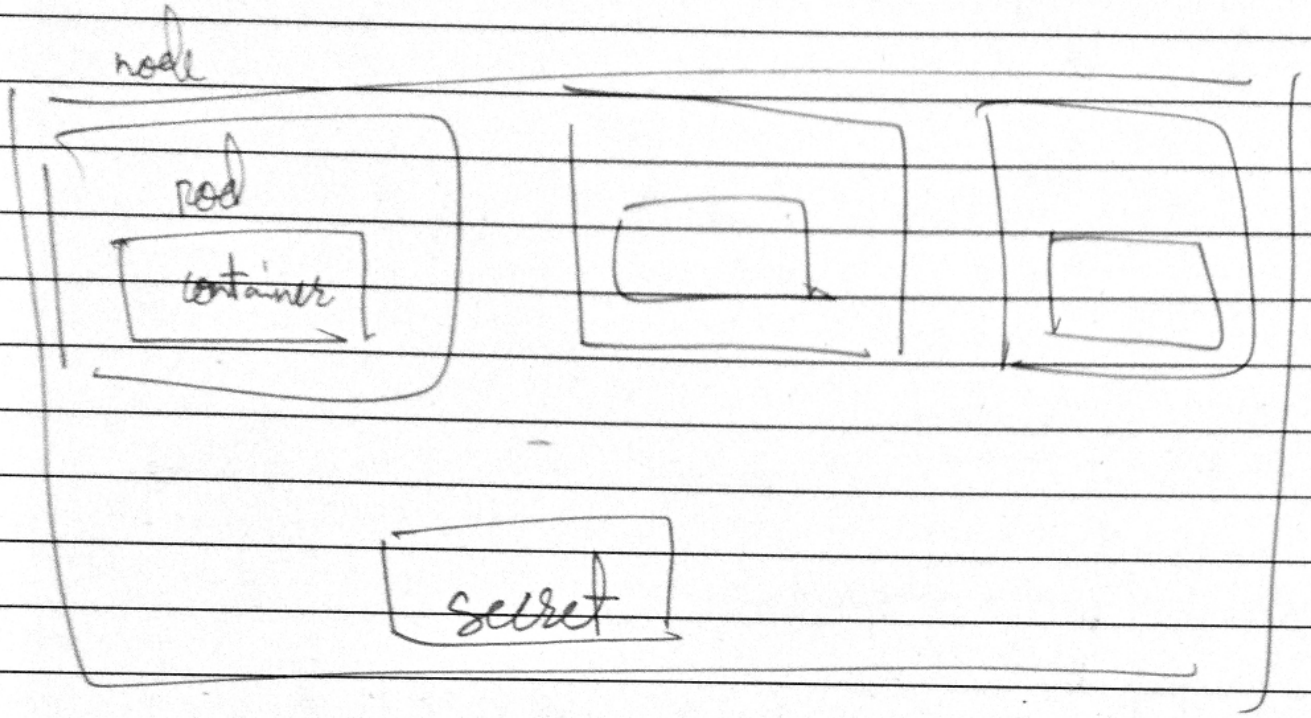
1. - lots of duplicate code
2. - really hard to picture the flow of events
3. - really hard to remember what properties an event should have
4. - really hard to test some event flows

solution

1. build central library NPM package
2. precisely define events in shared library
3. Typescript
4. write tests for as much as possible/reasonable

Date _____ 20 ____

Securely Storing Secrets w/ K8s



secret K8s object
stores key/value pairs
exposed as env variable

for Kubectl create secret generic <secret-name> --from-literal=
key=pair

interesting command → create object
declarative approach → config file

Shafjeel

Date _____ 20 ____

Microservices : Authentication Strategies & Options

Is this person logged in?

Fundamental option # 1

1.1 centralized auth service [OS] → [AS]

~~Sync req~~ direct req b/w microservices

if auth service goes down problems

1.2 auth gateway

[AS] → [OS]

Date _____ 20 ____

Fundamental option # 2

all services know how to authenticate a use

problem

rescinding access to user difficult after a valid token has been assigned

choose option # 2 because sticking w/ idea of independent services

short lived JWT / Cookies etc

Auth service Token refresh Login

either MS or client makes request to auth service for refresh

Date _____ 20 ____

event bus

rescind access event

each MS has short lived in memory store cache for lifetime of token

Cookies

- transport mechanism
- moves data b/w browser & server
- automatically managed by the browser

JWTs

- Authentication & Authorisation mechanism
- stores any data we want
- managed manually

Date _____ 20 ____

Microservices auth requirements

- tell us details about a user
- able to handle auth info
- must have built in tamper resistant way to expire or invalidate itself
- must be easily understood b/w languages & frameworks
- must not require some kind of backing data store on the server

Issues w/ JWTs & server side rendering

Only cookies can communicate on first page loads.

npm cookie-session
express-session

Date _____ 20 ____

if (!req.session || !req.session.jwt)

↓

equal to

↓

if (!req.session?.jwt)

Testing w/ MicroServices

What's the scope of our tests?

- test single piece of code in isolation → unit test
- test how different pieces of code work together
- test how different components (MS → DB, MS → Bus) work together
- test how different services work together

Date _____ 20 ____

testing goals

1, Basic request handling

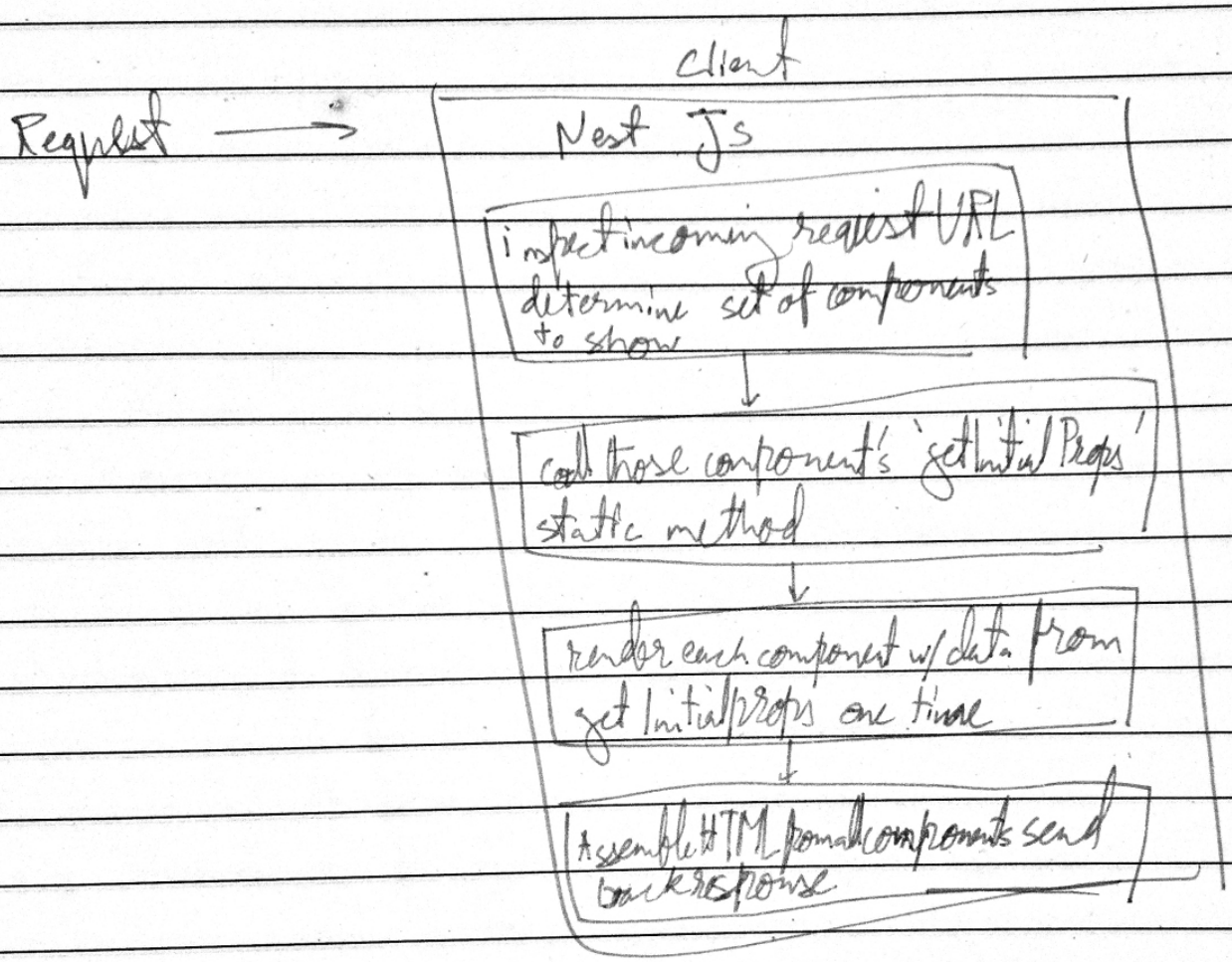
2, some test around mongo models

3, event emitting & recieving

Jest → - start in memory copy of MongoDB
- startup express app
- use supertest to make fake requests to our express app
- run assertions to make sure the request did the right thing

Date _____ 20 ____

Server Side Rendered Frontend - Next.js



getInitialProps deprecated use getServerSideProps

Date _____ 20 ____

Request Source

- Hard refresh of page
- Clicking link from different domain
- Typing URL in address bar

→ getServerSideProps executed on server

- navigating from one page to another within the app

→ getServerSideProps executed on client

Date _____ 20 ____

Code sharing & reuse b/w services

~~- cont~~

- common git repo submodules
- Npm package

Date _____ 20 ____

NATS streaming server

node-nats-streaming

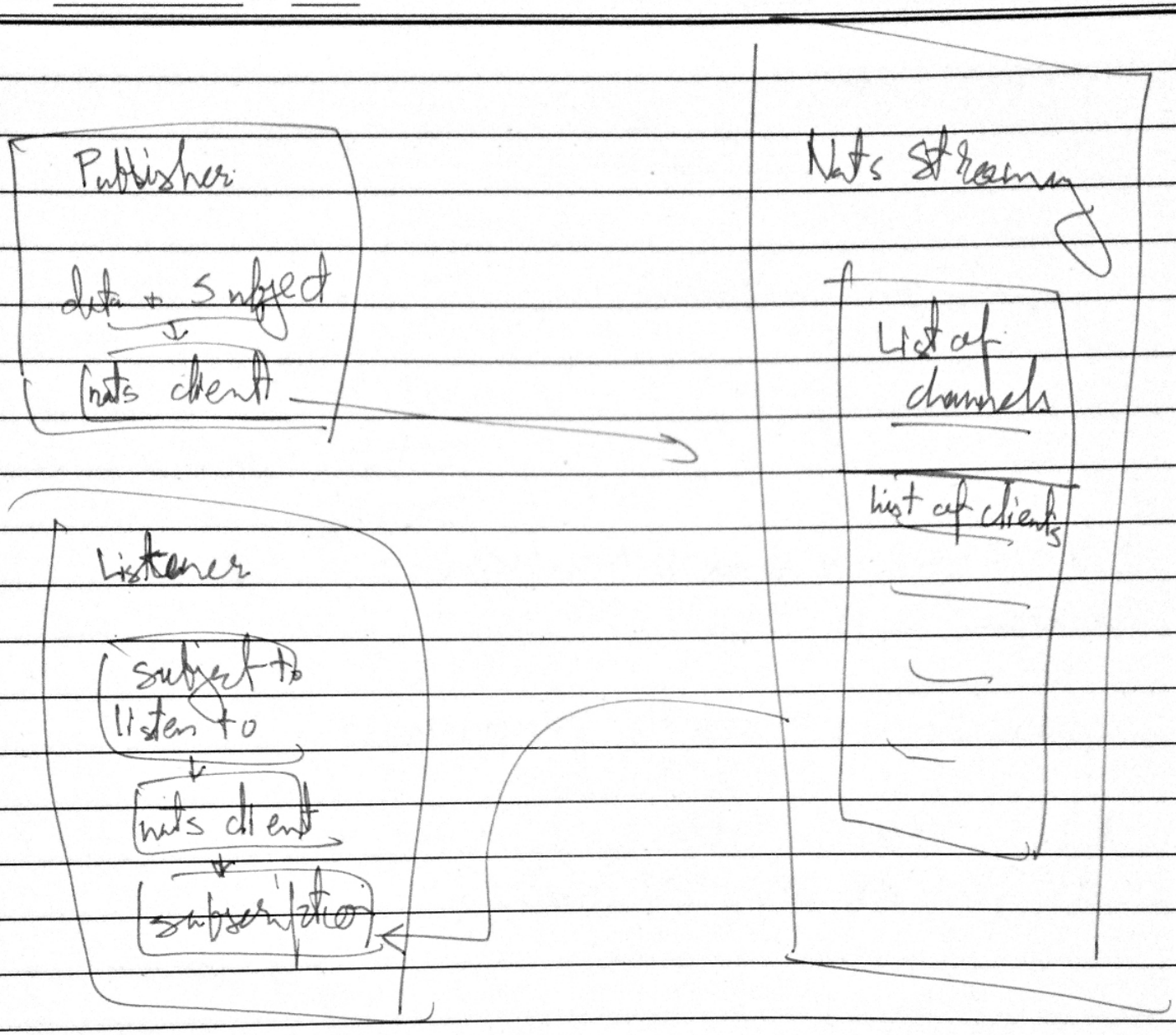
↑ client library for NATS streaming

Nats streaming requires us to subscribe to channels

Events emitted to specific channels

Nats Streaming stores all events in memory (default) flat files, or MySQL/PostgreSQL

Date _____ 20 ____



* nats can only take data as string (JSON event AKA message

Sharjeel

Date _____ 20 ____

queue group

created inside channel

listener join queue group

nats streaming randomly sends event to a single service in queue group

Default behavior when event recieved is marked Ack

Manual Ack

process message successfully then Ack
if not Ack in time resend to different listener in queue group / same listener

Date _____ 20 ____

Concurrency issues

- listener fails to process the event
- one listener might run more quickly than another
- Nats might think a client is still alive when it is dead
- we might receive the same event twice

Durable Subscription

subscription ~~has~~ with identifier

makes sure gets events while server is down

Date _____ 20 ____

Optimistic Concurrency Control

no locks, use versioning

before commit make sure no other transaction has modified data it has read

update if current