REDUX

Episode IV: A New Hope

Vi i SpareBank1 lager sabla mange skjermbilder. Hva består egentlig en frontendapplikasjon av?

DATA:

DATA:

kontoer
betalinger

DATA:

kontoer

betalinger

sortering

filter

skjemadata

DATA:

- kontoer
- betalinger

} app state

- sortering
- filter
- skjemadata

} view state

VIEW:

VIEW:

utseende

visningslogikk

ACTIONS

ACTIONS:

brukerinteraksjon forårsaker data-endring

DATA:

kontoer, betalinger } app state

sortering, filter, skjemadata } view state

VIEW:

utseende
visningslogikk

ACTIONS:

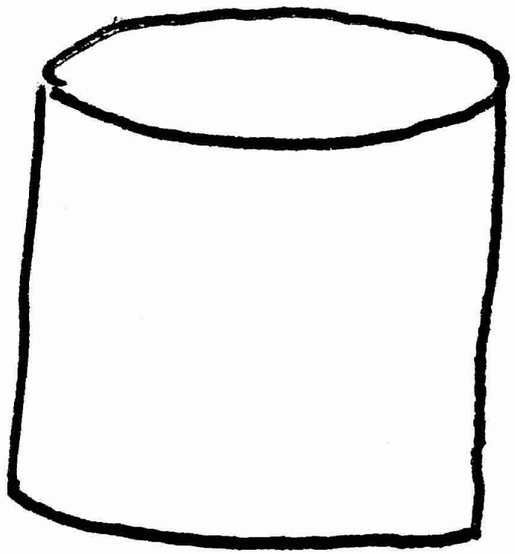
brukerinteraksjon
forårsaker data-endring

Data.

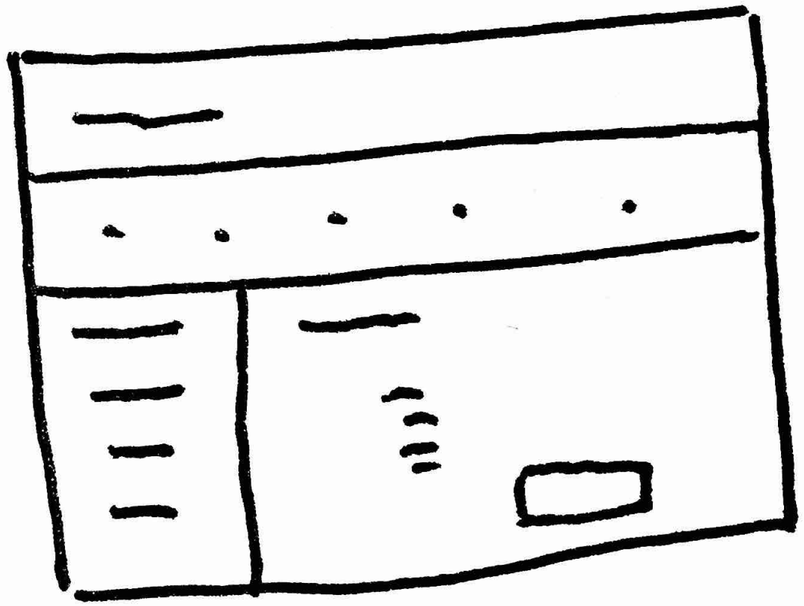
Views.

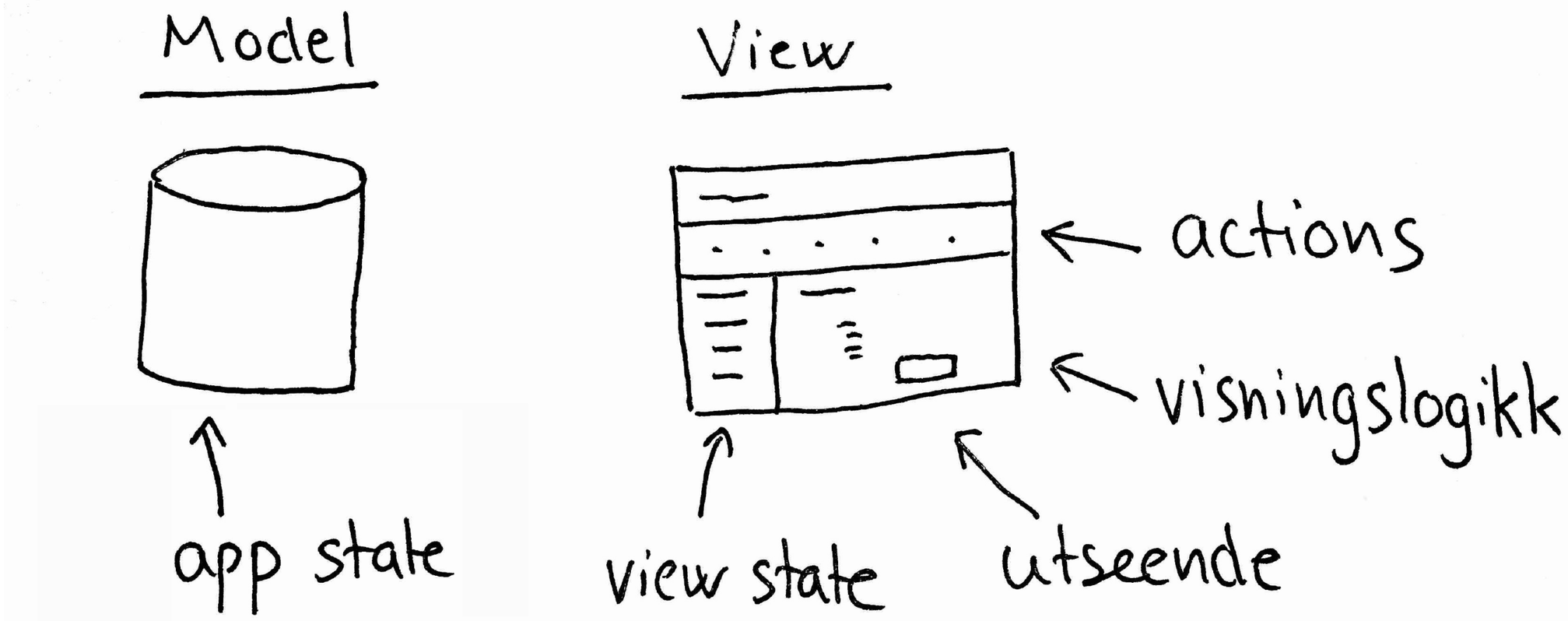
Actions.

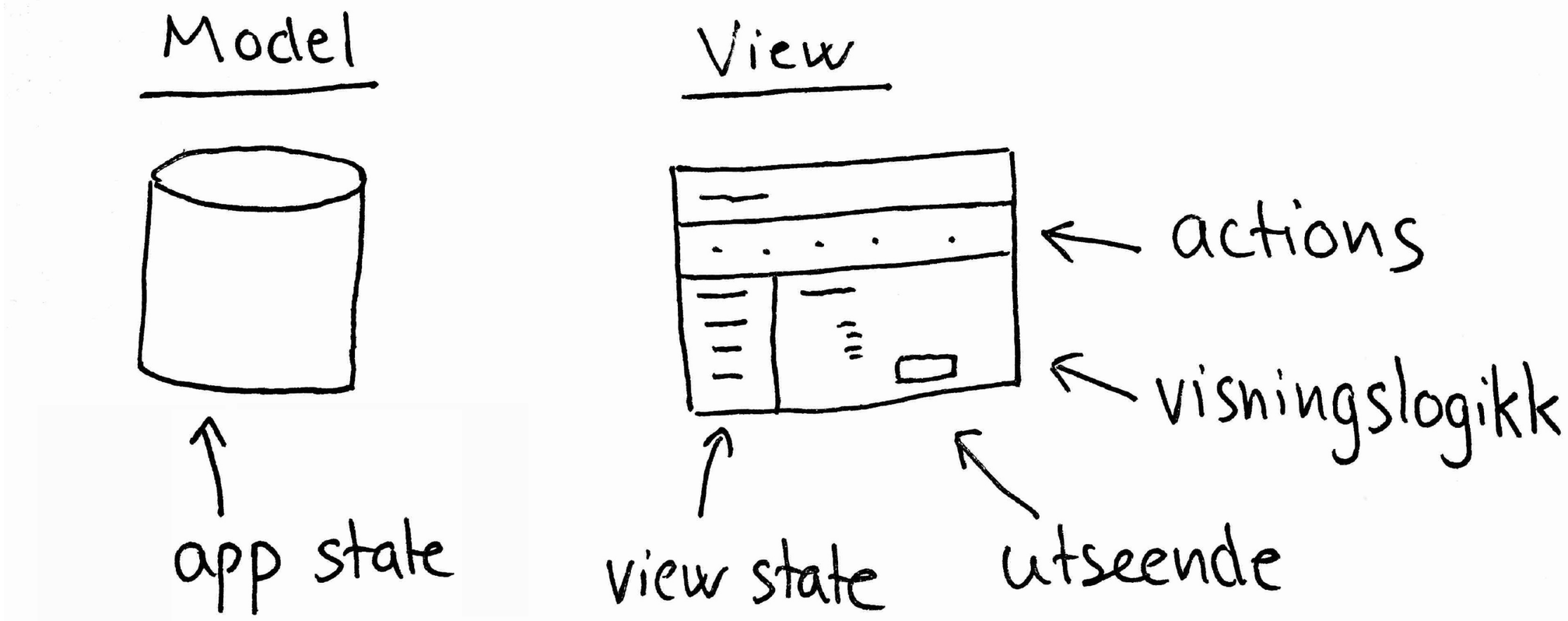
Model

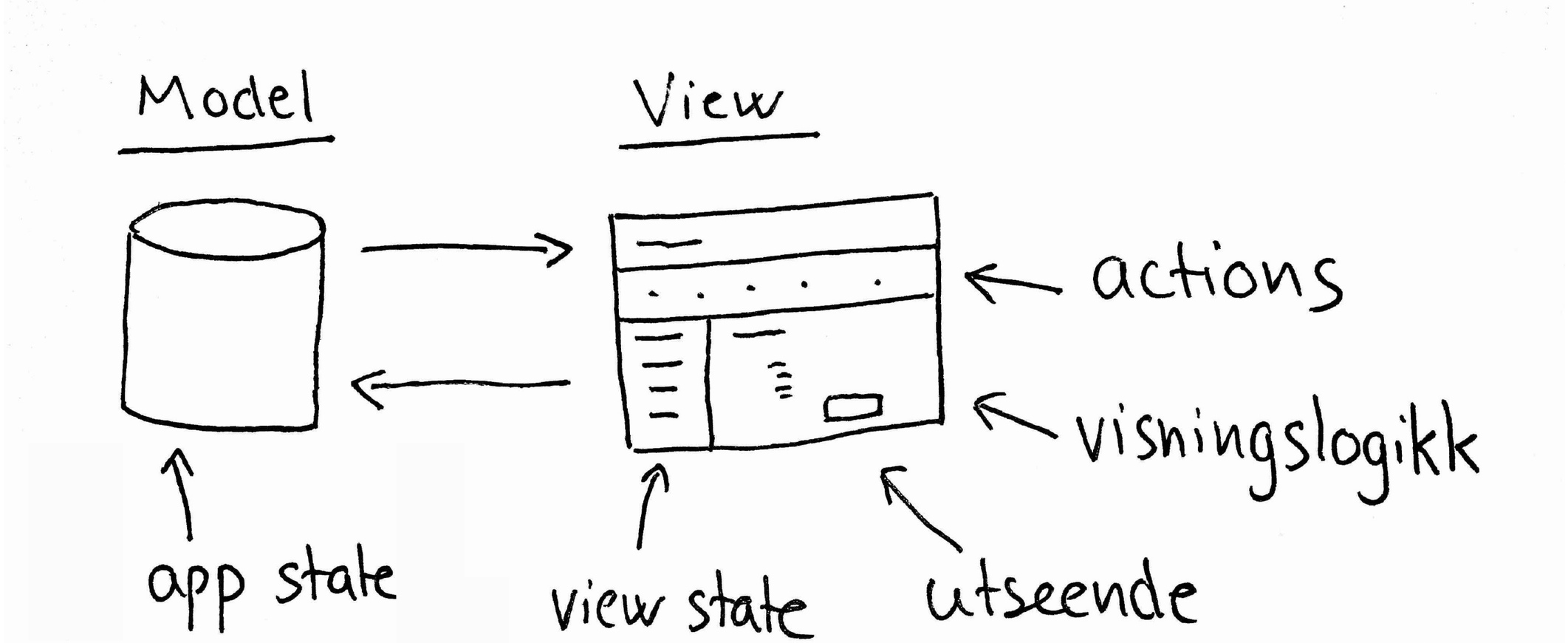


View

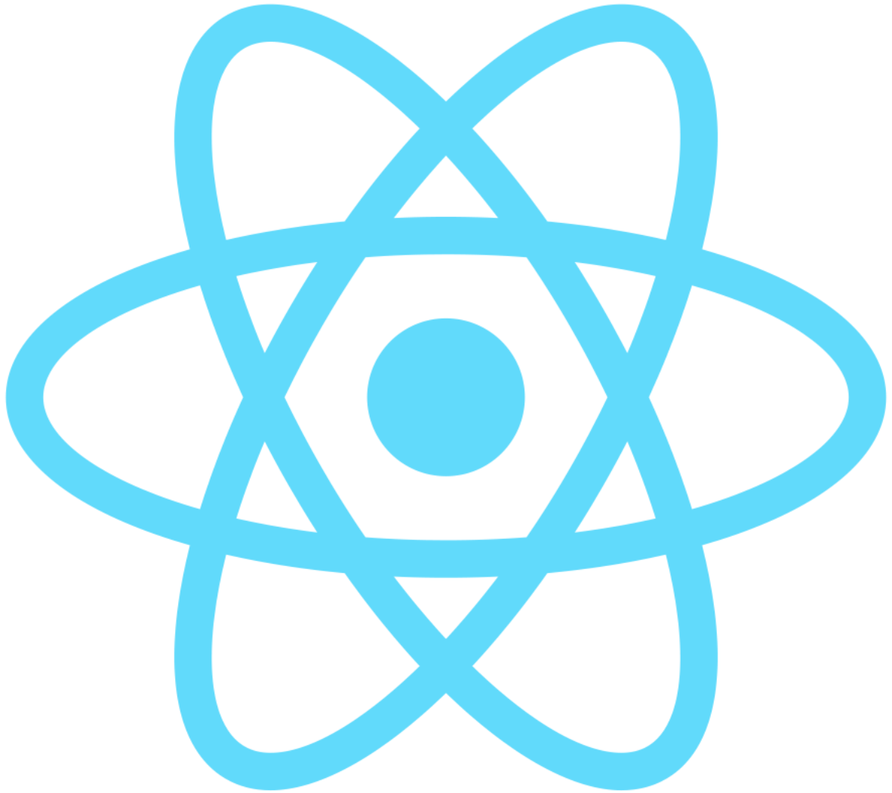


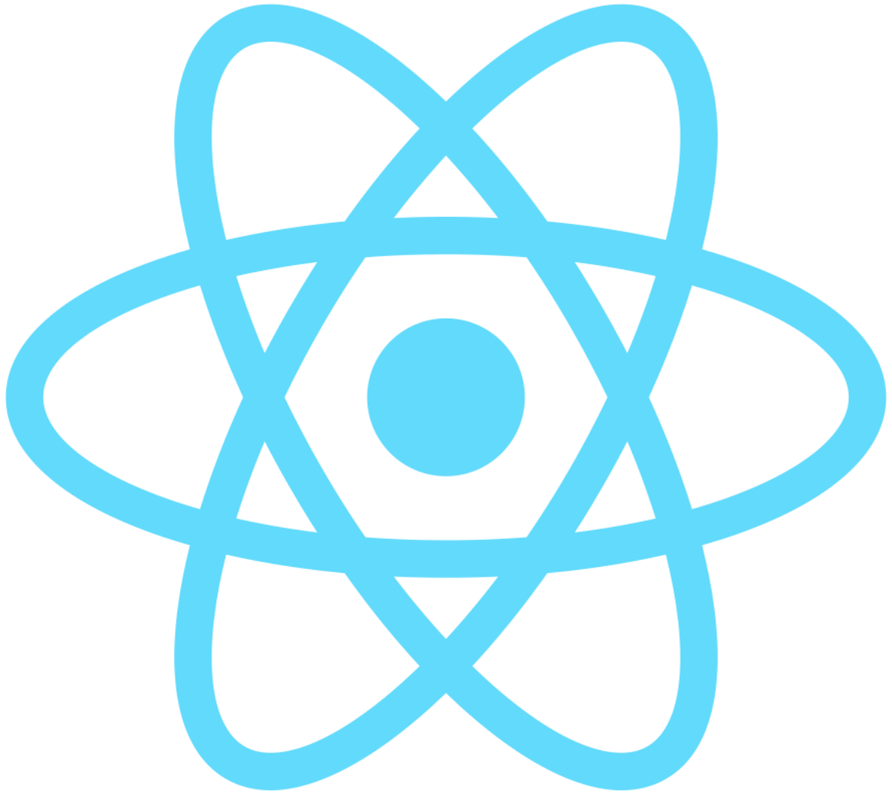

Model
View
actions
visningslogikk
app state
view state
utseende


Model
View
actions
visningslogikk
app state
view state
utseende


Model
View
actions
visningslogikk
app state
view state
utseende

En utfordring.
Bruker trykker på KID/Melding, deler av viewet skal endre seg.
Vi må enten 1) detaljert gå inn å endre de delene som nå skal oppdatere seg (errorprone, tidkrevende, skalerer dårlig) eller 2) re-rendre hele viewet (da mister vi f.eks de verdiene som brukeren har skrevet inn, hvis man lager et nytt tekstfelt).



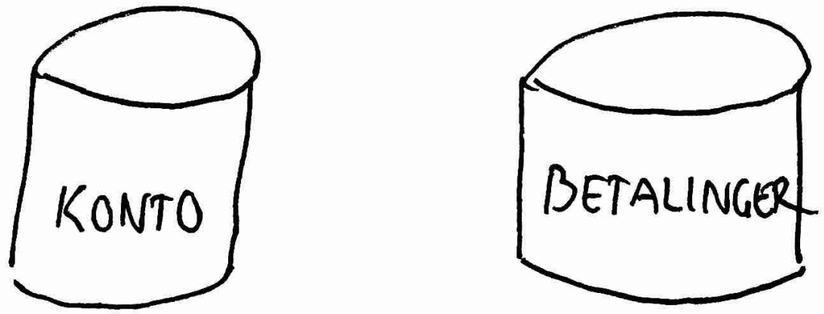



RE-RENDER
ALL THE THINGS
memegenerator.net


DATA
VIEW
HTML
(utseende)

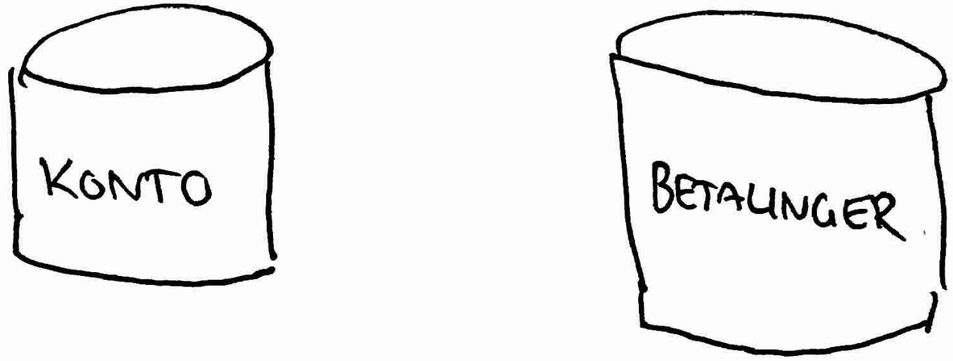
^Da har vi kutta to ansvarsområder fra viewene våre. App state er borte, handlingene er borte, men utseende og visningslogikk består.

Backbone



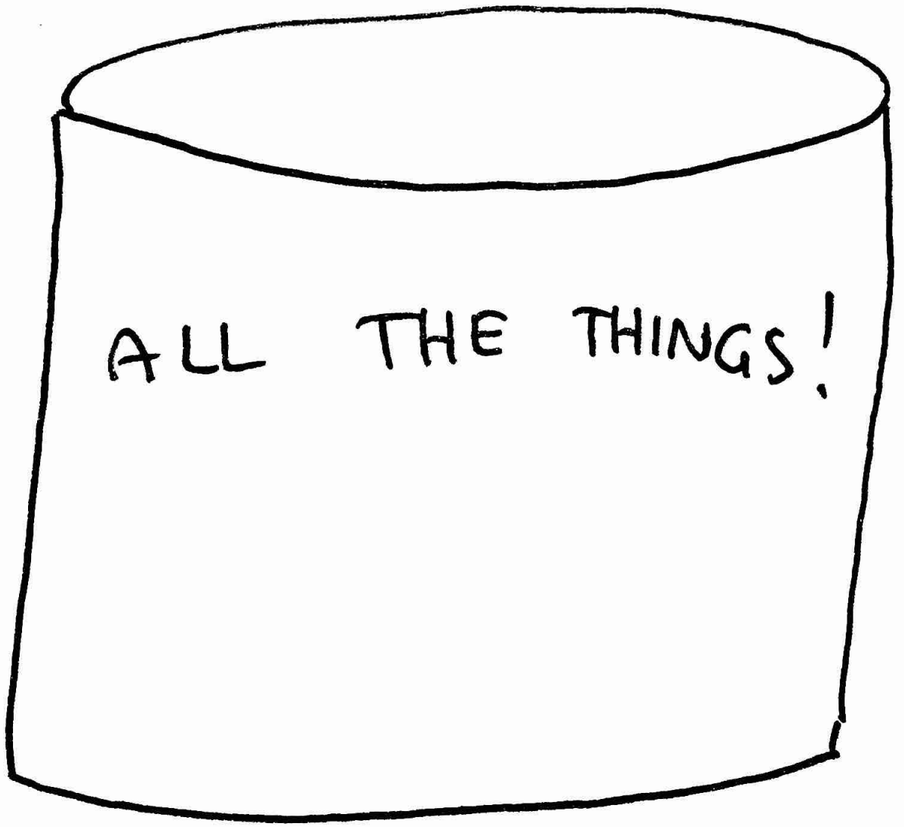
Models

Flux (facebook)

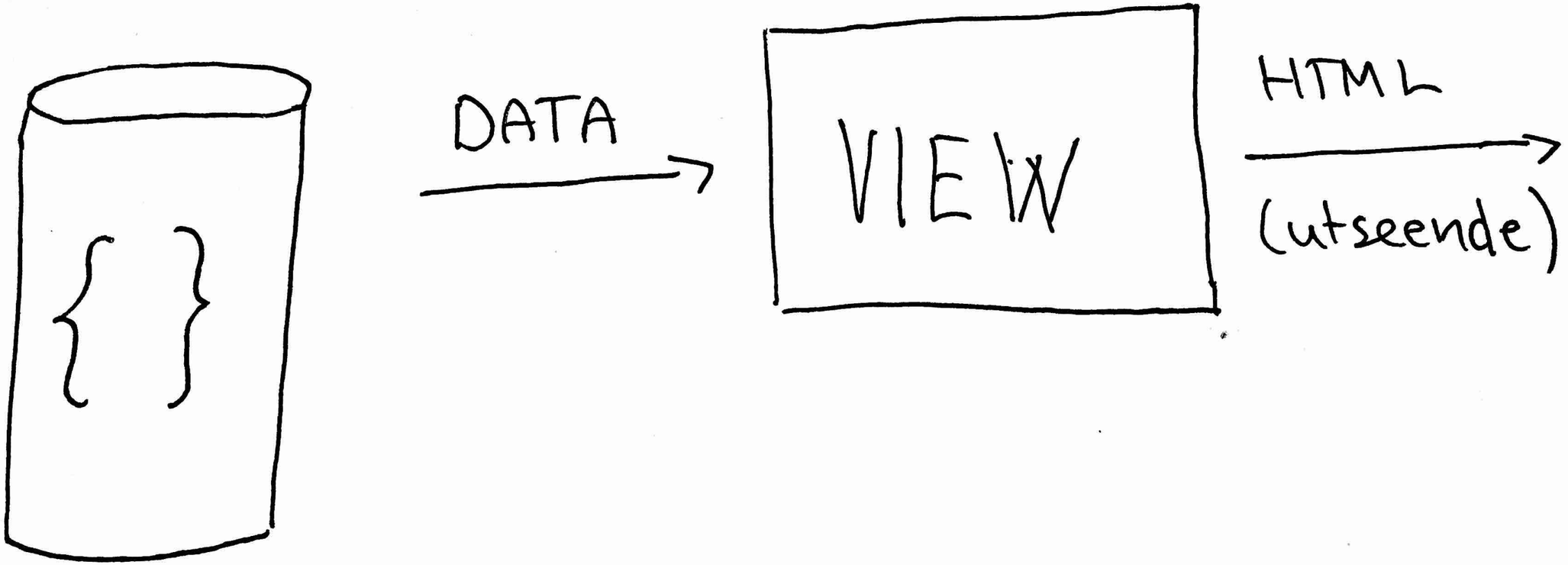


Stores

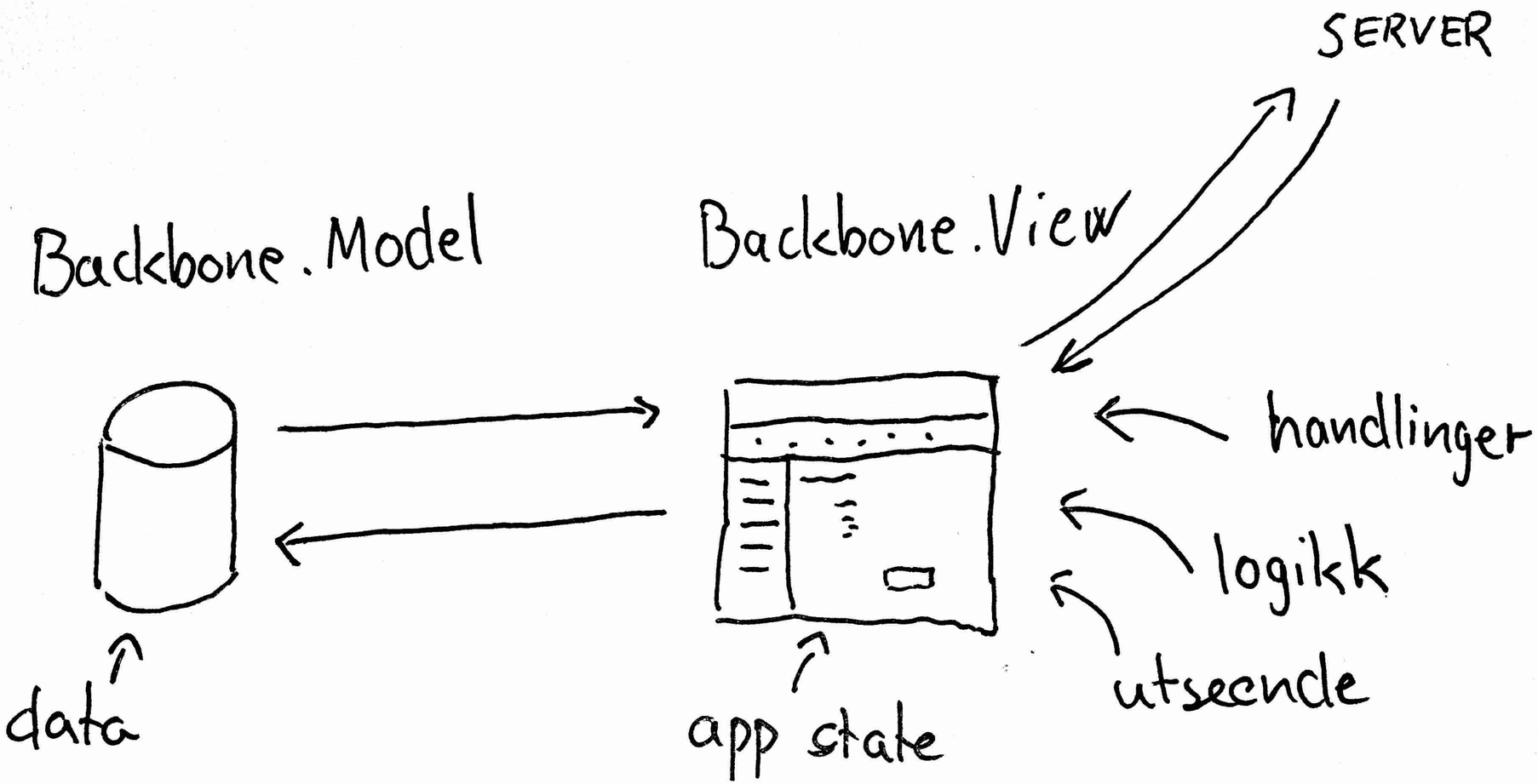
Redux

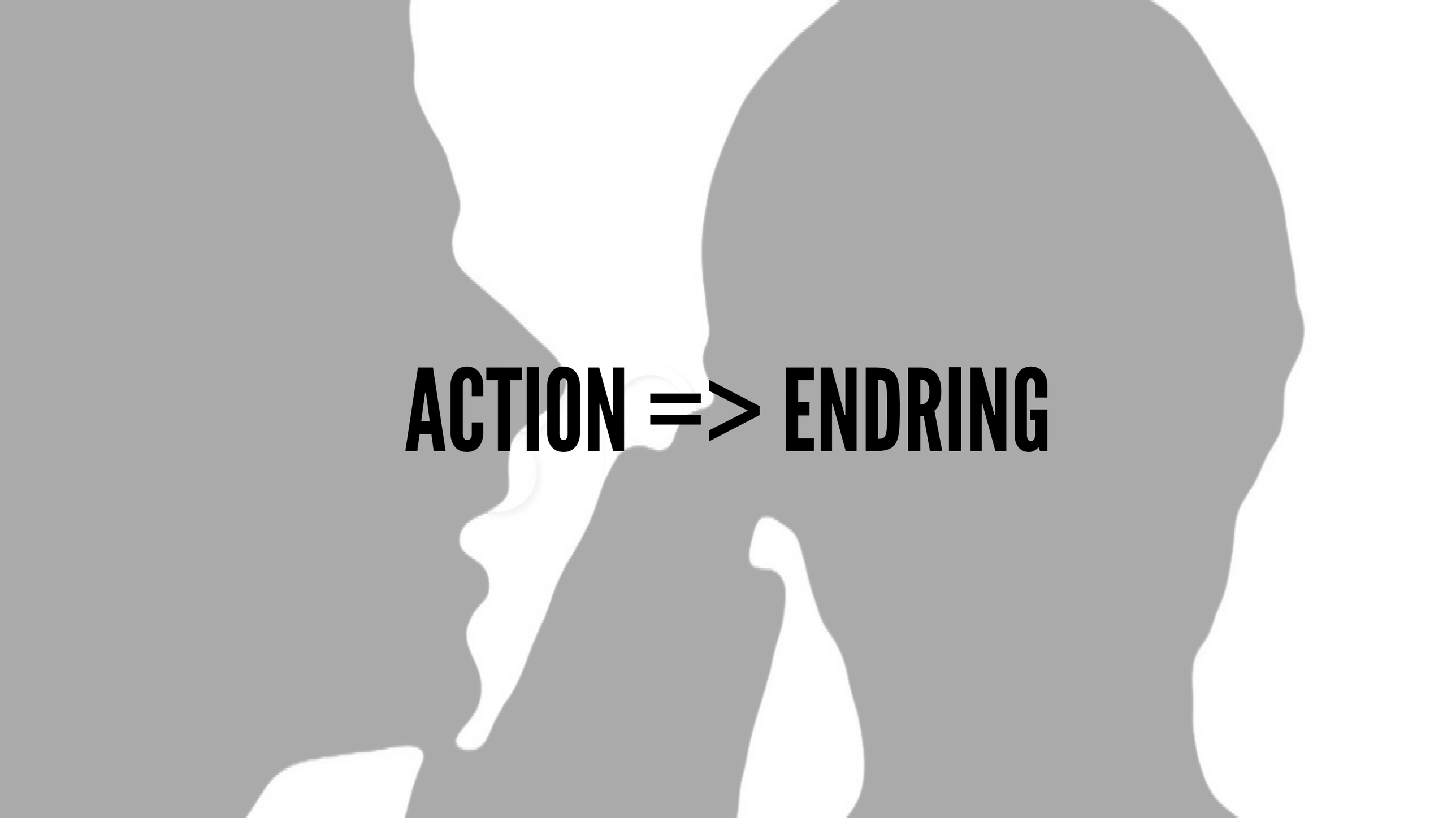


```
const state = {
    accounts: [...],
    creditCards: [...],
    debitCards: [...],
    savingAgreements: [...],
    userDetails: {...},
    ...
}
```


{ }
DATA
VIEW
HTML
(utseende)

Actions


SERVER
Backbone.Model
Backbone.View
handlinger
logikk
utsecncle
data
app state


ACTION => ENDRING

ACTION -> ENDRING

```
// action:
{
    type: 'DELETE_PAYMENT',
    paymentId: '123058572'
}
```

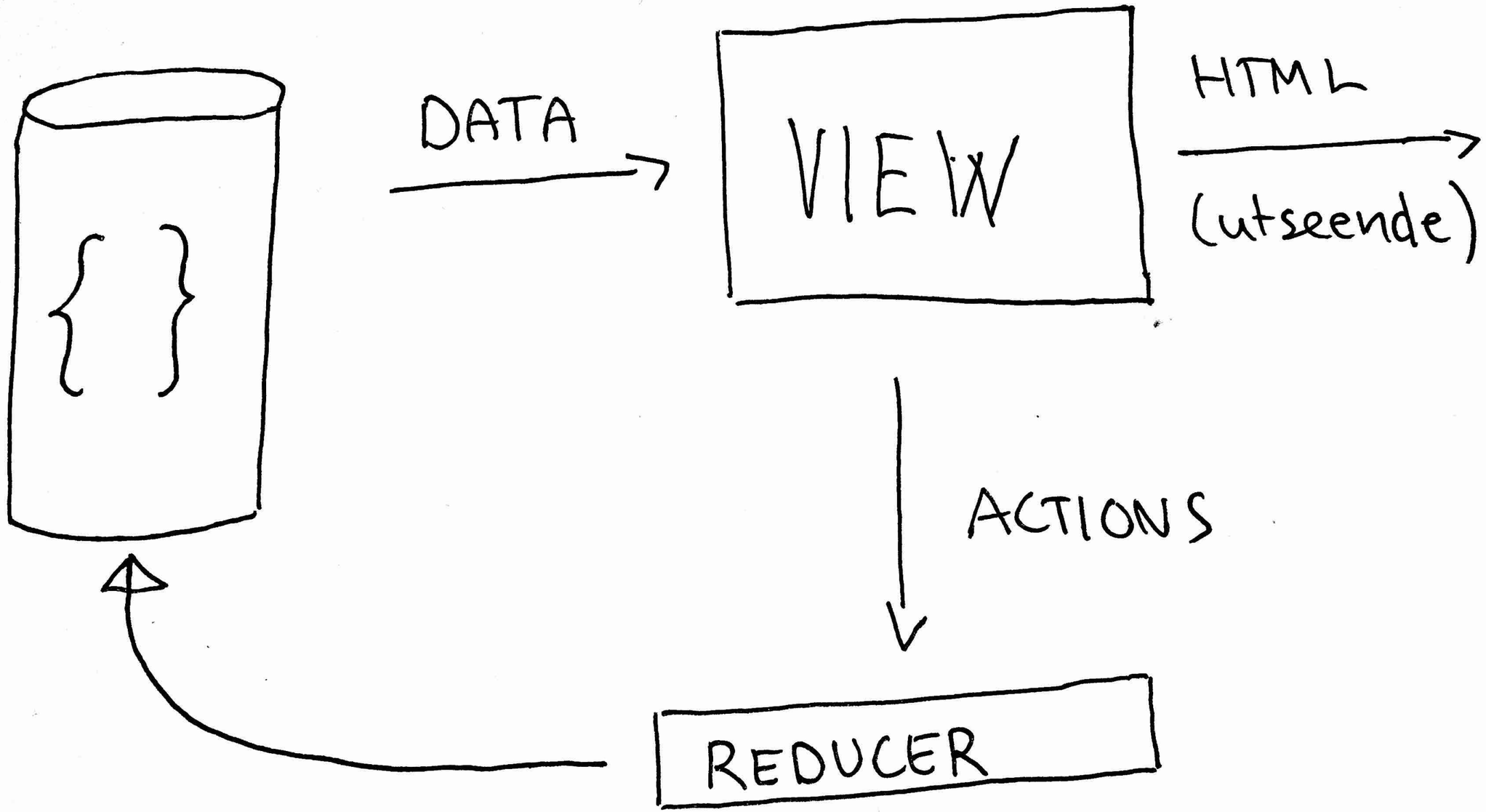
ACTION -> ENDRING

```
// action:
dispatch({
     type: 'DELETE_PAYMENT',
     paymentId: '123058572'
})
```

STATE, ACTION => STATE

STATE, ACTION => STATE

```
function deletePayment(state, action) {
    if (action.type === 'DELETE_PAYMENT') {
        // slett betaling
        return newState;
    }
}
```


{ }
DATA
VIEW
HTML
(utseende)
ACTIONS
REDUCER

REDUCERS:

Små funksjoner + testing = <3

elm

the best of functional programming in your browser

writing great code should be easy ... now it is

try or install

Hello HTML

```
main = span [class "welcome-message"] [text "Hello, World!"]
```

Writing HTML apps is super easy with start-app. Not only does it render extremely fast, it also quietly guides you towards well-architected code.

ARKITEKTURFORDELER:

- enkle funksjoner gjør endringer
- kun ett sted å spore for endringer
- hot reloading
- debug panel
- tydelig ansvarsfordeling
- testing er enklere

Spørsmål?

@ewndl