



# Miniprojekt 1 - Create React App

Grupp C: Filip Sunnemar, Fredrik Malmborg, Christian Ågren



# Överblick

1. Produkt
  - a. Mockup
  - b. Slutprodukt
2. Genomförande
  - a. Komponenter
  - b. React Router
  - c. API
  - d. Felhantering (error handling)
3. Resultat (DEMO)
4. Reflektion



# Produkt - Pokédex

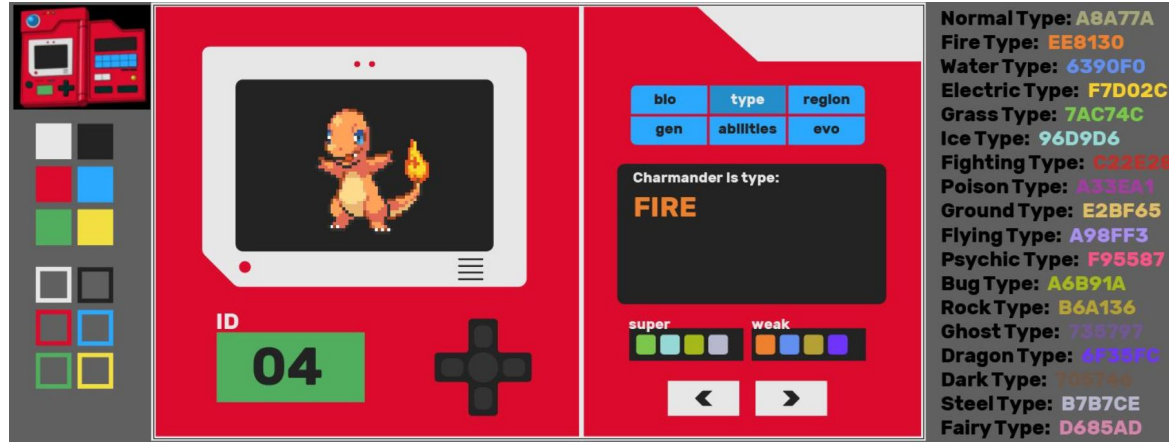
## Grundläggande funktionalitet:

- Söka på Pokémon och enkelt hitta dess information (Bio, vikt, höjd, typ, attacker, mm).
- Göra ett modernt, uppdaterat verktyg med mycket information...
- ... men behålla den grundläggande Pokémon charmen/stilen.

## Utökad funktionalitet (tillkom senare):

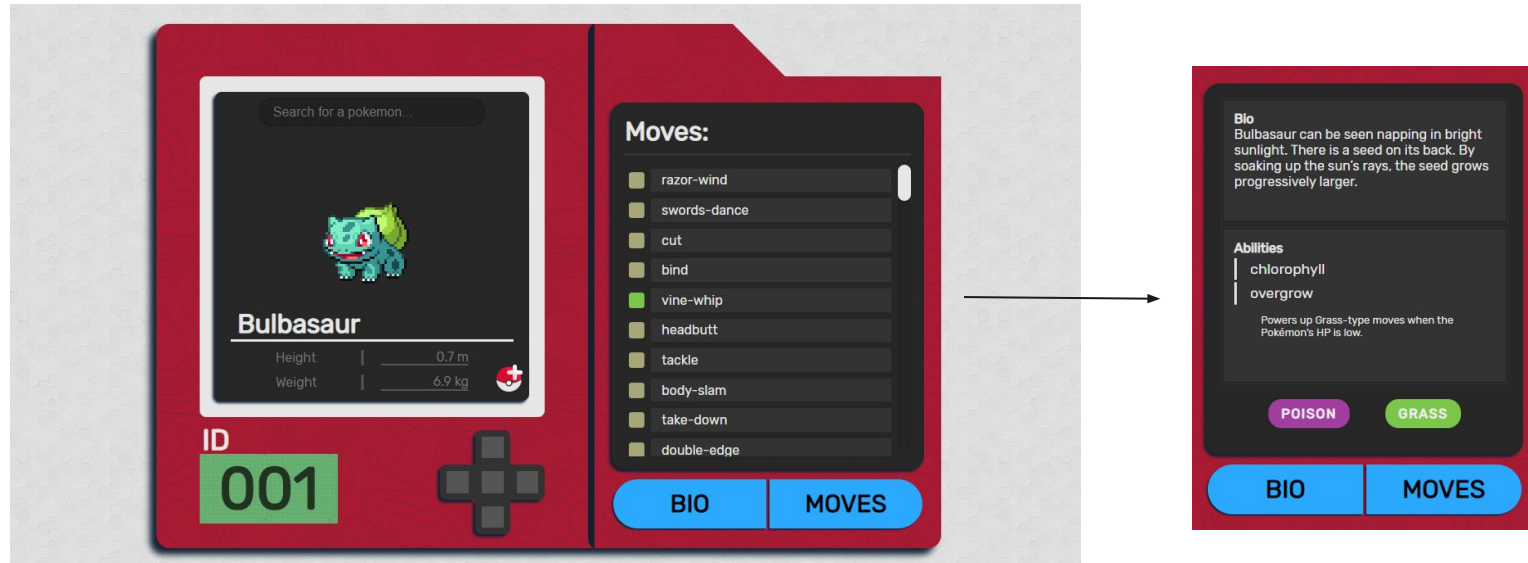
- Användaren ska kunna skapa sitt eget team med 6 pokemon, med skräddarsydda attacker och verktyget ska kunna säga vad teamet är starkt respektive svagt mot.

# Produkt - Pokédex



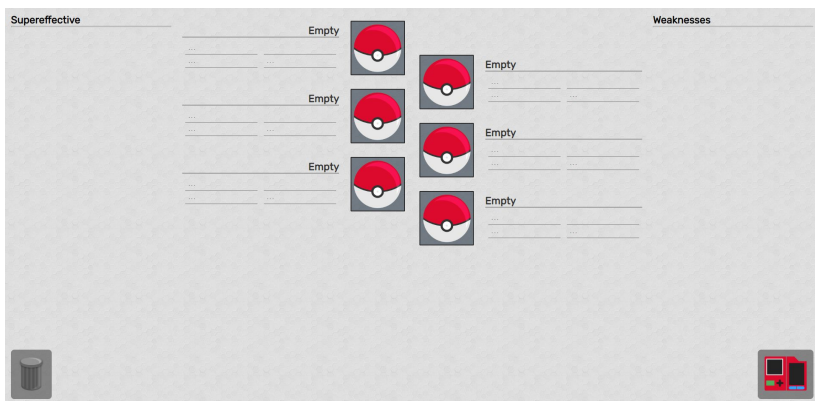
Första mockup av grundläggande funktionalitet

# Resultat - Pokédex

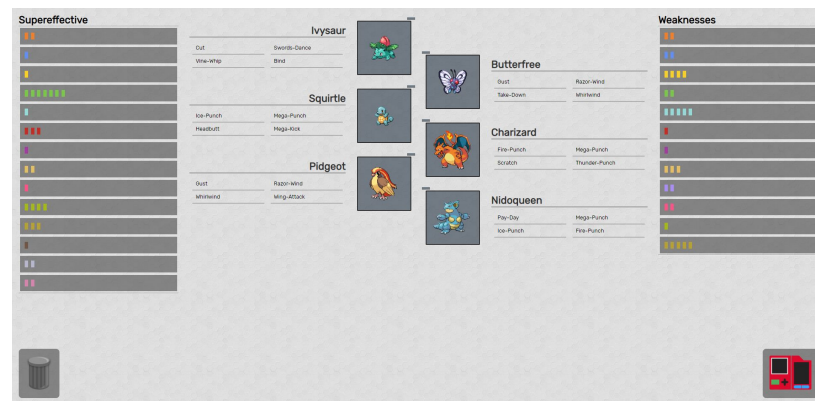
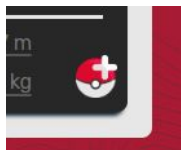


Huvudsidan för Pokédex

# Resultat - Pokédex



tomt



fullt

Teamsidan där man kan bygga sitt lag från  
Pokédexen (Inte MVP)




# Genomförande

## Generellt:

- Dagliga scrumliknande möten med eller utan kodning (Teams)
- Eget ansvar för egentilldelade områden/komponenter

## Deadlines:

- MVP: 9 mars
- Slutgiltig produkt: 17 mars



# Genomförande - Komponenter

## Målsättning:

- Återanvända funktionalitet
- Selfsustaining, fristående (felhantering samt introduktion i överliggande komponenter)
- Hellre fler än färre





# Genomförande - React Router

- Navigera mellan Pokédex och “team”-sidan
- Hålla koll på vilken Pokémon som man är inne på
- Hantera fel som t.ex visa en pokemon som inte finns

# Genomförande - API

- Hämtar önskad data från PokeAPI.co
- Axios - hämtar data
- async/await - asynkront
- React.lazy/Suspense - presenterar fallback komponent medan data laddas in



Hämtar först Pokemon i toppen av komponentträdet och hämtar sedan diverse mindre datapaket längre ner i de självständiga komponenterna för maximal självständighet och felhantering



# Genomförande - Felhantering

- Error boundaries
- try/catch kontrollerar om individuella fetches misslyckas
- Presenterar 404 sida (missingno) om första fetchningen misslyckas



**DEMO**



# Reflektion

## Med projektet:

### Positivt/Vad gick bra:

- Tydlig produktvision ✓
- Tidigt genomförd MVP ✓
- Bra kommunikation och samarbete ✓

### Negativt/Vad gick mindre bra:

- Hann inte klart med allt ✗
- Kunde haft bättre struktur ✗

## Med React:

### Positivt/Vad gick bra:

- Create React App ✓
- Enkelt att bygga en stor, avancerad applikation ✓
- Enkelt att dela upp arbetet ✓

### Negativt/Vad gick mindre bra:

- Animationer ✗
- React Router ✗
- Immutability gör att det krävs fler if statements ibland till väldigt mycket kod ✗