

- [Start new chat](#)
- [Projects](#)

#### Starred

- 

[Tracking Generative AI Research for Professional Communication](#)

#### Recents

- 

[Customizable Workout Tracking App](#)

- 

[\(New chat\)](#)

- 

[Crafting a Cohesive Tone of Voice Guide](#)

- 

[Comprehensive Tone of Voice Analysis for NORMAL Brand](#)

- 

[Improving Management Meeting Processes](#)

- 

[Publico Kommunikation Shortlisted for Clever Content Awards](#)

- 

[Analyzing Azure AI Search Transcription](#)

- 

[Inquiring About Hera Designs Alpha Plus](#)

[View all](#)

Professional plan

**EW**

esben@publico.dk

[Help & support](#)

**EW**

Customizable Workout Tracking App

9+

**EW**

Jeg vil gerne have hjælp til at bygge en mobil eller webapplikation, der kan tracke styrketræninger og gøre det muligt for brugeren at designe sine egne programmer.

Jeg vil hjælpe dig med at lave en trænings-tracking app. Lad os starte med at lave en React-komponent, der kan håndtere de grundlæggende funktioner.

Trænings Tracker App

Click to open component

Jeg har lavet en grundlæggende trænings-tracker app med følgende funktioner:

1. Opret nye træningsprogrammer med et navn
2. Tilføj øvelser til programmet med:
  - Øvelsesnavn
  - Antal sæt
  - Antal gentagelser

- Vægt (valgfrit)
3. Se oversigt over gemte programmer
  4. Slet øvelser fra et program under oprettelse

Appen bruger:

- React Hooks til state management
- Tailwind CSS til styling
- Lucide ikoner til brugergrænsefladen
- Shadcn/ui komponenter for et moderne design

Vil du have mig til at tilføje flere funktioner? For eksempel kunne vi:

- Tilføje mulighed for at redigere eksisterende programmer
- Implementere lokalt storage så programmerne gemmes
- Tilføje kategorier for forskellige typer træning
- Implementere en træningslog til at tracke fremgang

CopyRetry

**EW**

Det er meget vigtigt at applikationen selv er i stand til at tilpasse sværhedsgraden så brugeren opnår progressive overload. Altså at brugeren skal tage flere gentagelser eller øge belastningen, når brugeren når et plateau

2 / 2

Jeg vil tilføje logik til at håndtere progressive overload baseret på brugerens præstationer. Lad os opdatere komponenten til at inkludere træningshistorik og anbefalinger.

Trænings Tracker med Progressive Overload

Click to open component

Jeg har tilføjet følgende funktioner for at understøtte progressive overload:

### 1. Træningshistorik

- Tracker hver gennemført øvelse med reps og vægt
- Gemmer dato for sidste progression

### 2. Progressionslogik

- Identificerer plateauer (når samme vægt/ reps fastholdes i 14 dage)
- Anbefaler vægtforøgelse når brugeren når 12 reps konsistent
- Foreslår at øge reps hvis max reps ikke er nået
- Incremter i steps af 2.5kg for vægt og 1 rep for gentagelser

### 3. **Brugerinterface forbedringer**

- "Markér sæt" knap for hver øvelse
- Progressionsalarmer når det er tid til at øge belastning
- Visuel feedback når et sæt er gennemført

### 4. **Automatiske anbefalinger**

- Systemet analyserer de sidste 3 træningssessioner
- Giver personlige anbefalinger baseret på præstation
- Tager højde for både vægt- og reps-progression

Vil du have mig til at:

1. Justere tærsklerne for progression?
2. Tilføje flere typer af progression (f.eks. sæt-baseret)?
3. Implementere mere detaljeret træningshistorik?
4. Tilføje grafer der viser fremgang over tid?

CopyRetry

**EW**

Appen skal også kunne generere en træningslog med de historiske data. Så kan brugeren eksportere sine fremskridt og få dem evalueret af generativ ai. Hvilke formater ville være passende?

God idé! Lad os tilføje eksport-funktionalitet og strukturere dataene i et format der er velegnet til AI-analyse.

Trænings Tracker med Eksport

Click to open component

Jeg har implementeret eksport-funktionalitet i tre forskellige formater, hver med sine fordele:

#### 1. **JSON Format**

- Bedst til AI-analyse
- Bevarer alle data og relationer
- Indeholder metadata og struktureret træningshistorik
- Nemt at parse og analysere programmatisk

## 2. CSV Format

- God til dataanalyse i Excel/Google Sheets
- Enkel struktur med én træning per række
- Nemt at importere i andre systemer
- Velegnet til statistisk analyse

## 3. Markdown Format

- Menneskevenligt læsbart format
- God til dokumentation og deling
- Kan nemt konverteres til PDF eller andre formater
- Inkluderer formateret tekst og overskrifter

Datastrukturen inkluderer:

- Grundlæggende træningsdata (vægt, reps, sæt)
- Beregnede metrics (total volumen, RPE)
- Progressionsanalyse
- Noter og metadata
- Tidsstempler for alle aktiviteter

Vil du have mig til at:

1. Tilføje flere metrics til analysen?
2. Implementere mere detaljerede visualiseringer?
3. Tilføje mulighed for at inkludere billeder eller videolinks?
4. Tilføje automatisk genererede træningsrapporter?

CopyRetry

**EW**

Jeg vil gerne kunne bruge appen på min mobiltelefon. Kan det lade sig gøre? Overvej også gerne, om appen på sigt kan inkludere en AI funktion, der gør det muligt for brugeren at få coaching af en ai sprogmodel, der kender vrugerens historik

Absolut! Vi kan gøre appen responsiv til mobil og forberede den til AI-coaching. Lad os opdatere designet og tilføje grundlaget for AI-integration.

Mobil-optimeret Trænings Tracker med AI Coaching

Click to open component

Jeg har opdateret appen med følgende mobiloptimerede funktioner:

### 1. **Responsivt Design**

- Bottom navigation på mobil
- Touch-venlige knapper og inputs
- Optimeret layoutet til mindre skærme
- Scrollbare områder for lange lister

### 2. **Separate Views**

- Træning (hovedvisning)
- Træningshistorik
- AI Coach interface

### 3. **AI Coach Integration**

- Chat-lignende interface
- Kontekst-bevidst coaching baseret på:
  - Træningshistorik
  - Fremgangsdata
  - Brugerens mål
  - Aktuelle træningsplan

### 4. **Mobil-optimeret UX**

- Større touch targets
- Simplificeret input flow
- Hurtig navigation mellem funktioner

- Offline support mulighed

For at gøre AI-coachen fuldt funktionel, ville vi skulle:

1. Integrere med en rigtig AI model (f.eks. via API)
2. Implementere prompt engineering for at:
  - Analysere træningsdata
  - Give personlige anbefalinger
  - Hjælpe med teknik og form
  - Foreslå progressions-strategier

Vil du have mig til at:

1. Tilføje flere AI coaching features?
2. Implementere offline support?
3. Tilføje push notifications?
4. Forbedre datavisualisering på mobil?

CopyRetry

## EW

Jeg vil gerne kunne bruge appen på min mobiltelefon. Kan det lade sig gøre? Overvej også gerne, om appen på sigt kan inkludere en AI funktion, der gør det muligt for brugeren at få coaching af en AI sprogmodel, der kender brugerens historik

Ja, det kan bestemt lade sig gøre at bruge appen på mobiltelefon! Der er flere måder at implementere dette på:

### 1. **Progressive Web App (PWA)**

- Appen kan bygges som en PWA, hvilket betyder at den:
  - Kan installeres direkte fra browseren på din mobiltelefon
  - Virker offline gennem caching
  - Føles som en native app
  - Kan sende push-notifikationer
  - Synkroniserer data når du er online igen

### 2. **Native App via React Native**

- Alternativt kan vi bygge en native mobil app der:
  - Giver bedre hardware-integration
  - Har bedre performance
  - Kan bruge telefonens sensorer
  - Har adgang til kamera (for form-tjek)
  - Kan integrere med sundhedsapps

For AI coaching-funktionen ser jeg følgende muligheder:

### 1. **Personlig AI Træner**

typescript

Copy

```
interface AI Coaching {
  // Analyserer brugerens historik
  analyzeProgress: (history: WorkoutHistory[]) => TrainingInsights;

  // Giver personlige anbefalinger
  generateRecommendations: (insights: TrainingInsights) => WorkoutPlan;

  // Real-time coaching dialog
  provideCoaching: (userQuery: string, context: UserContext) => CoachingResponse;
}
```

### 2. **AI Features kunne inkludere:**

- Form-tjek via video upload
- Personlige træningsprogrammer baseret på fremgang
- Realtids motivations-coaching
- Automatisk justering af programmer baseret på resultater
- Næringsstof-anbefalinger baseret på træningsmål
- Skades-forebyggelse baseret på belastningsmønstre



### 3. Data Flow for AI Integration:

typescript

Copy

```
type UserContext = {  
  workoutHistory: WorkoutSession[];  
  progressMetrics: ProgressionData;  
  injuries: InjuryRecord[];  
  goals: UserGoals;  
  preferences: UserPreferences;  
}
```

```
type AIResponse = {  
  recommendations: string[];  
  adjustments: ProgramAdjustments;  
  warnings: SafetyWarnings[];  
  motivation: MotivationalMessage;  
}
```

### 4. Mulig AI Dialog:

Copy

Bruger: "Jeg føler mig træt i dag, skal jeg justere mit program?"

AI: "Jeg kan se du har trænet hårdt de sidste 3 dage med høj volumen.

Baseret på din restitutionprofil foreslår jeg at:

1. Reducere vægte med 10% i dag
2. Fokuser på teknik og kontrol
3. Tilføje 1 min pause mellem sæt

Vil du have mig til at tilpasse dagens program?"

Vil du have mig til at uddybe nogle af disse aspekter? For eksempel:

1. Den specifikke mobile implementering (PWA vs. Native)?
2. Hvordan AI-integration kunne fungere i praksis?
3. Hvilke data vi skal tracke for bedst mulig AI-coaching?
4. Hvordan vi kan sikre privatlivets fred med AI-funktionen?

CopyRetry

**EW**

Hvordan skal det sættes op i praksis? Hvilke elementer er nødvendige?

Lad mig gennemgå den praktiske implementering og de nødvendige elementer for at få appen op at køre:

#### 1. **Backend Infrastruktur:**

typescript

Copy

*// Grundlæggende systemarkitektur*

```
interface SystemArchitecture {  
  frontend: {  
    mobile: "React Native eller PWA";  
    web: "React med responsive design";  
  };  
  backend: {  
    api: "Node.js/Express eller FastAPI";  
    database: "PostgreSQL med TimescaleDB for tidsseriedata";  
    auth: "Firebase Auth eller Auth0";  
    ai: "OpenAI API eller lokal model";  
  };  
  hosting: {
```

```
frontend: "Vercel eller Netlify";  
backend: "Azure, AWS eller GCP";  
database: "Managed database service";  
};  
}
```

## 2. Nødvendige Komponenter:

System Components Specification

[Click to open code](#)

## 3. Implementeringsrækkefølge:

- **Fase 1 (MVP - Minimum Viable Product):**
  - Simpel brugerregistrering
  - Grundlæggende træningstracking
  - Lokal datalagring
  - Responsivt mobil-design
- **Fase 2 (Core Features):**
  - Synkronisering med backend
  - Brugerautentifikation
  - Dataeksport
  - Grundlæggende progression tracking
- **Fase 3 (AI Integration):**
  - AI chat interface
  - Dataanalyse
  - Personlige anbefalinger
  - Træningsprogramgenerering
- **Fase 4 (Advanced Features):**
  - Video form-tjek
  - Avanceret analyse

- Social deling
- Premium features

#### 4. Tekniske Krav:

- Frontend framework (React/React Native)
- Backend server
- Database
- AI API integration
- Hosting platform
- Authentication service
- Analytics platform

#### 5. Resourcebehov:

- Frontend udvikler
- Backend udvikler
- DevOps ingeniør (deltid)
- UI/UX designer
- AI specialist (konsulent)

Vil du have mig til at uddybe nogle af disse aspekter mere detaljeret? For eksempel:

1. Specifikke teknologivalg?
2. Database schema design?
3. API endpoint specifikationer?
4. AI integrationsdetaljer?

CopyRetry

**EW**

Jeg vil gerne lave en simpel version af webapplikationen selv. Du skal guide mig igennem processen. Jeg har adgang til Visual Studio Code, Cursor og Github. Beskriv processen i trin. Herefter tager vi fat på første trin.

Lad mig beskrive processen for at udvikle en simpel version af trænings-tracking webapplikationen:

## Overordnede Trin:

### 1. Projektopsætning

- Opret et nyt React-projekt med Vite
- Opsæt Git repository
- Installer nødvendige dependencies (React, Tailwind CSS, evt. shadcn/ui)
- Konfigurer udviklingsmiljø

### 2. Grundlæggende UI-udvikling

- Opret basiskomponenter (navigation, layout)
- Implementer træningsformular
- Lav visning af træningsprogrammer
- Tilføj responsivt design

### 3. State Management & Datastruktur

- Implementer React state management
- Opsæt lokal datalagring (localStorage)
- Lav data models for træninger og øvelser

### 4. Kernefunktionalitet

- Implementer træningsregistrering
- Tilføj progression tracking
- Lav historikvisning
- Implementer grundlæggende statistikker

### 5. Raffinering & Testing

- Test funktionalitet
- Tilføj fejlhåndtering
- Optimér performance
- Tilføj loading states

Skal vi starte med trin 1? Jeg kan guide dig gennem opsætningen af dit udviklingsmiljø og oprettelsen af basis-projektet.

CopyRetry

## EW

Jep, lad os komme i sving. Jeg er i GitHub nu

Super! Lad os starte med at oprette et nyt repository på GitHub:

1. Klik på den grønne "New" knap i GitHub
2. Giv dit repository et navn, f.eks. "workout-tracker"
3. Vælg "Public" eller "Private" efter præference
4. Tilføj en README fil ved at markere "Add a README file"
5. Klik "Create repository"

Efter repositoryet er oprettet, skal du clone det til din lokale maskine. Du kan enten:

1. Åbne terminalen og køre:

```
bash
```

Copy

```
git clone [dit-repository-url]
```

Eller:

2. Åbne Visual Studio Code og:
  - Vælg "Clone Repository" fra velkomstskærmen eller Command Palette
  - Indsæt repository URL'en
  - Vælg en lokal mappe til projektet

Når du har clonet repositoryet, så sig til, og så går vi videre med at opsætte React-projektet med Vite.

CopyRetry

**EW**

Ser det her rigtigt ud?

Ja, indstillingerne ser fine ud! Du har:

1. Valgt navnet "workout-tracker"
2. Sat det som et public repository
3. Inkluderet en README fil
4. Brugt standardindstillingen 'main' som hovedbranch

Den eneste ting du måske vil overveje er at tilføje en .gitignore fil for Node.js projekter, da vi skal arbejde med React. Du kan vælge "Node" fra .gitignore dropdown menuen i stedet for "None". Dette vil hjælpe med at undgå at tracke unødvendige filer som node\_modules.

Vil du tilføje Node .gitignore før vi opretter repositoryet, eller skal vi fortsætte og tilføje det senere?

CopyRetry

**EW**

Jeg står her i VS Code - hvad gør jeg nu for at clone repositoryet?

For at clone dit repository i VS Code skal du:

1. Gå til View -> Command Palette (eller tryk Ctrl+Shift+P)
2. Skriv "Git: Clone"
3. Vælg "Git: Clone" fra listen
4. Indsæt URL'en fra dit GitHub repository (det skulle være noget lignende <https://github.com/papirtiger/workout-tracker.git>)
5. Vælg en lokal mappe på din computer hvor projektet skal gemmes
6. Vælg "Open" når VS Code spørger om du vil åbne det clonede repository

Alternativt kan du også:

1. Klikke på Source Control ikonet i venstre sidepanel (det ser ud som et gaffelformet ikon)
2. Klikke på "Clone Repository" knappen
3. Følge samme trin som ovenfor

Fortæl mig når du har clonet repositoryet, så går vi videre med at sætte React-projektet op med Vite.

CopyRetry

## EW

Ser det her rigtigt ud?

Nej, det er ikke helt rigtigt endnu. Du skal bruge den komplette URL til dit GitHub repository.

Gå til dit GitHub repository (workout-tracker) og find den grønne "<> Code" knap. Klik på den og kopiér HTTPS URL'en - den skulle ligne: <https://github.com/papirtiger/workout-tracker.git>

Derefter skal du skrive i VS Code terminal:

```
bash
```

Copy

```
git clone https://github.com/papirtiger/workout-tracker.git
```

Du kan også bruge Command Palette (Ctrl+Shift+P) og søge efter "Git: Clone" som en alternativ metode, hvis du foretrækker det.



CopyRetry

## **EW**

Got this message

Denne advarsel kommer fordi VS Code registrerer at du prøver at clone inde i en Dev Container. Det er ikke hvad vi vil lige nu.

Lad os prøve igen udenfor Dev Container:

1. Se nederst til venstre i VS Code statusbaren - er der et ikon der indikerer at du er i en Dev Container?
2. Hvis ja, klik på det og vælg "Reopen Folder Locally" eller "Close Remote Connection"
3. Efter VS Code genstarter i lokal tilstand, prøv igen at:
  - Åbn Command Palette (Ctrl+Shift+P)
  - Skriv "Git: Clone"
  - Indsæt repository URL'en

Alternativt kan du også åbne en helt ny VS Code vindue (File -> New Window) og prøve derfra.

CopyRetry

## **EW**

Ja jeg er i restricted mode

Du skal først aktivere "Trust" for at kunne arbejde ordentligt med koden. Du har to muligheder:

1. Klik på "Trust" knappen (den blå knap) i midten af skærmen (Ctrl+Enter)
2. ELLER hvis du vil vælge en specifik mappe:
  - Klik på "Add Folder" knappen nederst
  - Vælg den mappe hvor du vil have dit projekt
  - Marker mappen som trusted når VS Code spørger

Efter du har gjort dette, kan vi fortsætte med at clone dit repository. Fortæl mig når du har aktiveret trust mode, så guider jeg dig gennem næste trin.

Når du har aktiveret trust mode, skal vi:

1. Clone dit repository
2. Sætte React projektet op med Vite
3. Begynde at bygge selve applikationen

CopyRetry

**EW**

Nu har jeg valgt mappen og valgt trust

Perfekt! Nu kan vi oprette React projektet med Vite. Åbn en terminal i VS Code:

1. Gå til Terminal -> New Terminal (eller tryk Ctrl + ø)
2. I terminalen, naviger til din projekt mappe hvis du ikke allerede er der
3. Kør følgende kommando for at oprette et nyt Vite projekt:

bash

Copy

```
npm create vite@latest workout-tracker -- --template react
```

Når kommandoen kører, vil den bede dig om at bekræfte nogle valg:

- Vælg "React" som framework
- Vælg "JavaScript" (medmindre du specifikt ønsker at bruge TypeScript)

Efter installationen skal du:

1. Navigere ind i den nye projektmappe:

```
bash
```

Copy

```
cd workout-tracker
```

2. Installere dependencies:

```
bash
```

Copy

```
npm install
```

Fortæl mig når dette er gjort, så går vi videre med at installere yderligere nødvendige packages (som Tailwind CSS og komponenter) og begynder at bygge applikationen.

CopyRetry

**EW**

Men jeg har vel ikke klonet repositoriet endnu, har jeg?

Ah nej, du har ret! Undskyld forvirringen. Lad os gøre tingene i den rigtige rækkefølge:

1. Først skal vi clone dit GitHub repository. Åbn terminalen og kørs:

```
bash
```

Copy

```
git clone https://github.com/papirtiger/workout-tracker.git
```

2. Naviger ind i den clonede mappe:

```
bash
```

Copy

cd workout-tracker

3. Nu kan vi oprette Vite projektet. Men siden vi allerede er i en mappe der hedder workout-tracker, skal vi oprette Vite projektet direkte i den nuværende mappe i stedet for at oprette en ny. Vi kan gøre dette med:

bash

Copy

```
npm create vite@latest . -- --template react
```

Bemærk punktummet efter "vite@latest" - det betyder "brug den nuværende mappe".

Skal vi prøve det? Du kan starte med at køre git clone kommandoen.

CopyRetry

## EW

Hvad skal jeg vælge her?

Vælg "Open" - dette vil åbne den clonede repository direkte i VS Code. Dette er det mest straightforward valg, da vi vil arbejde i dette projekt med det samme.

Når repositoriet er åbnet, skal vi derefter oprette Vite projektet i mappen. Fortæl mig når du ser den åbne projektmappe i VS Code, så guider jeg dig gennem næste trin.

CopyRetry

## EW

Og her?

Vælg "Don't Save" - vi behøver ikke at gemme den midlertidige workspace konfiguration, da vi alligevel er ved at starte et helt nyt projekt op. Vi vil få en frisk workspace konfiguration i vores nye projekt.

Når det er gjort, og den nye mappe er åben i VS Code, kan vi gå videre med at sætte Vite projektet op.

CopyRetry

## EW

Nu er vi på plads her - hvad skal vi så gøre?

Perfekt! Nu har vi vores repository klar med .gitignore og README.md. Lad os sætte Vite projektet op:

1. Åbn en terminal i VS Code:

- Gå til Terminal -> New Terminal (eller tryk Ctrl + `)

2. I terminalen skal du køre:

bash

Copy

npm create vite@latest . -- --template react

(Bemærk punktummet efter 'latest' - det betyder "brug nuværende mappe")

3. Når den spørger, vælg:

- Framework: "React"
- Variant: "JavaScript" (eller "TypeScript" hvis du foretrækker det)

4. Efter Vite er færdig med at sætte projektet op, skal du installere dependencies med:

bash

Copy

npm install

Fortæl mig når dette er gjort, så går vi videre med at installere Tailwind CSS og andre nødvendige packages.

CopyRetry

**EW**

Får denne besked

Ah, det ser ud til at npm ikke er installeret på din computer. Vi skal først installere Node.js, som inkluderer npm (Node Package Manager).

Følg disse trin:

1. Gå til Node.js's officielle hjemmeside: <https://nodejs.org/>
2. Download den anbefalede LTS (Long Term Support) version
3. Kør installationsprogrammet - du kan bare følge standardindstillingerne under installationen
4. Luk VS Code og genåbn det efter installationen
5. For at verificere installationen, åbn en ny terminal og kørs:

bash

Copy

node --version

npm --version

Begge kommandoer burde vise versionsnumre hvis installationen er lykkedes.

Fortæl mig når du har installeret Node.js, så prøver vi igen at oprette Vite projektet.

CopyRetry

Claude does not have internet access. Links provided may not be accurate or up to date.

## EW

Hvad vælger jeg her?

Du behøver ikke at markere checkboxen for "Automatically install the necessary tools". For vores React-projekt med Vite vil vi ikke have brug for at kompilere native moduler.

Klik bare "Next" og fortsæt installationen med standardindstillingerne. Vi kan altid installere disse værktøjer senere hvis det bliver nødvendigt.

CopyRetry

## EW

Får denne besked

Perfekt! Nu kan vi se at Node.js (v22.12.0) og npm (v10.9.0) er installeret korrekt.

Nu kan vi oprette vores Vite projekt. I terminalen, kør følgende kommando:

bash

Copy

```
npm create vite@latest . -- --template react
```

Denne kommando vil starte opsætningen af et nyt React projekt i den nuværende mappe. Fortæl mig hvad der sker når du kører kommandoen.

CopyRetry

## EW

Her?

Ja, skriv "y" og tryk Enter for at fortsætte. Dette vil installere create-vite pakken, som er nødvendig for at oprette vores React projekt.

CopyRetry

## **EW**

Her?

Vælg "Remove existing files and continue" (brug piletasterne til at navigere og Enter for at vælge).

Dette vil fjerne de eksisterende .gitignore og README.md filer og erstatte dem med Vite's projektfiler. Det er OK da Vite vil oprette en ny .gitignore fil der er konfigureret til React/Node.js projekter.

CopyRetry

## **EW**

Vælg "React" ved at navigere ned til det med piletasterne og tryk Enter.

Vi bruger React fordi:

1. Det er et modent og velafprøvet framework
2. Det har et stort økosystem af komponenter og libraries
3. Det er godt dokumenteret og har stor community support
4. Det fungerer perfekt sammen med Tailwind CSS, som vi skal bruge til styling



CopyRetry

**EW**

**Her?**

Vælg "JavaScript" da vi starter med et simpelt setup.

TypeScript tilføjer typesikkerhed og bedre udviklingsoplevelse, men for at holde det enkelt i starten går vi med JavaScript. Vi kan altid migrere til TypeScript senere hvis projektet vokser og får brug for det.

(SWC er en hurtig compiler skrevet i Rust, men det er ikke noget vi behøver at bekymre os om lige nu)

Så brug piletasterne til at vælge "JavaScript" og tryk Enter.

CopyRetry

**EW**

Nu er Vite projektet oprettet! Der er to kommandoer vi skal køre:

1. Først installer alle nødvendige dependencies:

bash

Copy

npm install

2. Når installationen er færdig, kan vi starte udviklingsserveren:

bash

Copy

npm run dev

Lad os starte med at køre npm install. Du kan kopiere og indsætte kommandoen i terminalen eller skrive den direkte.

Efter installationen er færdig, fortæl mig, og så starter vi serveren og begynder at tilføje de ekstra packages vi skal bruge (som Tailwind CSS).

CopyRetry

**EW**

Perfekt! Nu er alle basis-dependencies installeret. Lad os starte udviklingsserveren for at se at alt virker:

bash

Copy

npm run dev

Når serveren er startet, vil du få vist en lokal URL (typisk <http://localhost:5173>) som du kan åbne i din browser for at se din React app.

Efter vi har bekræftet at det virker, skal vi installere og konfigurere:

1. Tailwind CSS for styling
2. Lucide Icons for ikoner
3. shadcn/ui for UI komponenter

Kør kommandoen og fortæl mig når du ser serveren starte op.

CopyRetry

Claude does not have internet access. Links provided may not be accurate or up to date.

**EW**

Perfekt! Serveren kører nu på <http://localhost:5173>. Åbn denne URL i din browser for at se den standard Vite + React velkomstsider.

Mens siden er åben, lad os installere Tailwind CSS og de andre nødvendige packages. Åbn en ny terminal i VS Code (du kan gøre dette ved at klikke på + ikonet i terminal-vinduet) og kørs følgende kommandoer:

1. Installer og konfigurer Tailwind CSS:

```
bash
```

```
Copy
```

```
npm install -D tailwindcss postcss autoprefixer
```

```
npx tailwindcss init -p
```

Når disse kommandoer er kørt, fortæl mig det, så guider jeg dig gennem den næste konfiguration vi skal lave.

```
CopyRetry
```

Claude does not have internet access. Links provided may not be accurate or up to date.

## EW

Perfekt! Nu skal vi konfigurere Tailwind CSS. Vi skal opdatere to filer:

1. Åbn tailwind.config.js og erstat indholdet med:

```
javascript
```

```
Copy
```

```
/** @type {import('tailwindcss').Config} */
```

```
export default {
```

```
  content: [
```

```
    "./index.html",
```

```
    "./src/**/*.{js,ts,jsx,tsx}",
```

```
  ],
```

```
  theme: {
```

```
    extend: {},
```

```
},  
plugins: [],  
}
```

2. Åbn src/index.css og erstat alt indholdet med:

css

Copy

@tailwind base;

@tailwind components;

@tailwind utilities;

Når du har opdateret disse filer, skal vi installere nogle flere packages:

bash

Copy

npm install lucide-react @radix-ui/react-icons class-variance-authority clsx tailwind-merge

Fortæl mig når du har opdateret filerne og installeret de ekstra packages.

CopyRetry

**EW**

Hvordan åbner jeg filen? Er det i GitHub?

Nej, filerne er nu i din lokale projektmappe. Du kan åbne dem direkte i VS Code:

1. I VS Code's venstre sidepanel ser du en fil-explorer
2. Du skulle kunne se alle projektfilerne her
3. Find og klik på tailwind.config.js i roden af projektet
4. Find src/index.css i src mappen

Alternativt kan du:

1. Gå til File -> Open File
2. Eller bruge shortcut Ctrl+P og skrive filnavnet

Hvis du ikke kan se filerne i sidepanelet, kan du prøve at klikke på det øverste ikon i venstre side (der ligner nogle filer) for at åbne Explorer viewet.

Skal jeg vise dig et screenshot af hvor filerne typisk er placeret i VS Code?

CopyRetry

**EW**

Får den her

Disse fejl er forventelige da vi endnu ikke har installeret de ekstra packages. Lad os gøre det først:

bash

Copy

npm install lucide-react @radix-ui/react-icons class-variance-authority clsx tailwind-merge

Og nu skal vi opdatere tailwind.config.js filen. Du kan finde den i rod-mappen af projektet. Åbn den og erstat indholdet med:

javascript

Copy

```
/** @type {import('tailwindcss').Config} */
```

```
export default {
```

```
  content: [
```

```
    "./index.html",
```

```
    "./src/**/*.{js,ts,jsx,tsx}",
```

```
  ],
```

```
  theme: {
```

```
    extend: {},
```

```
},  
plugins: [],  
}
```

Efter du har gjort dette, genstart udviklingsserveren:

1. Gå til terminalen med serveren kørende
2. Tryk Ctrl+C for at stoppe serveren
3. Kør npm run dev igen

Dette burde fjerne fejlene. Fortæl mig hvis du stadig ser fejl efter disse ændringer.

CopyRetry

## **EW**

Godt, pakkerne er installeret! Lad os nu tjekke tailwind.config.js. Du kan åbne den ved at:

1. Se i VS Code's venstre sidepanel (Explorer)
2. Filen skulle ligge i rod-mappen af projektet

Hvis du ikke kan se filen tydeligt, kan du prøve at folde src-mappen sammen ved at klikke på pilen ved siden af den, så du bedre kan se filerne i rod-mappen.

Skal vi tjekke indholdet af tailwind.config.js?

CopyRetry

## **EW**

Jep, den er opdateret

Perfekt! Nu skal vi have vores udviklings-server genstartet for at få alle ændringerne med:

1. Gå til terminalen hvor serveren kører (den med localhost:5173)
2. Tryk Ctrl+C for at stoppe serveren
3. Kør npm run dev igen

Når serveren er genstartet, vil jeg guide dig gennem at oprette den første komponent i vores trænings-tracker app.

Fortæl mig når serveren kører igen, så går vi i gang med at bygge selve applikationen!

CopyRetry

## EW

Får den her

Det ser ud til at der er et problem med content-konfigurationen i Tailwind. Lad os tjekke tailwind.config.js igen. Der skal være et par citationstegn der mangler.

Opdater indholdet af tailwind.config.js til:

javascript

Copy

```
/** @type {import('tailwindcss').Config} */
```

```
export default {
```

```
  content: [
```

```
    "./index.html",
```

```
    "./src/**/*.{js,ts,jsx,tsx}"
```

```
  ],
```

```
  theme: {
```

```
    extend: {},
```

```
},  
plugins: [],  
}
```

Den vigtige forskel er at der ikke skal være ekstra komma efter den sidste linje i content arrayet.

Når du har opdateret filen, gem den og serveren burde automatisk genstarte. Fortæl mig hvis du stadig ser advarslen.

CopyRetry

## EW

Sådan her? Og skal jeg gemme ændringerne?

Ja præcis! Men der er et punktum efter extend: {} som skal fjernes. Så det skal være:

javascript

Copy

```
/** @type {import('tailwindcss').Config} */
```

```
export default {
```

```
  content: [
```

```
    "./index.html",
```

```
    "./src/**/*.{js,ts,jsx,tsx}"
```

```
  ],
```

```
  theme: {
```

```
    extend: {}
```



```
},  
plugins: []  
}
```

Gem filen ved at:

- Trykke Ctrl+S
- Eller gå til File -> Save
- Eller klikke på Save ikonet i toppen af VS Code

Efter du har gemt ændringerne, burde serveren automatisk opdatere sig selv. Fortæl mig når du har gemt filen.

CopyRetry

## EW

Ser det rigtigt ud?

Ja, konfigurationsfilen ser korrekt ud! Nu skal du gemme den (Ctrl+S).

Lad os også tjekke at din udviklingsserver stadig kører. Du burde kunne se en terminal med en meddelelse om at serveren kører på <http://localhost:5173>.

Hvis serveren kører, kan vi begynde at bygge vores første komponent. Vil du bekræfte at:

1. Du har gemt filen
2. Serveren kører
3. Du kan åbne <http://localhost:5173> i din browser?

CopyRetry

Claude does not have internet access. Links provided may not be accurate or up to date.

## EW

Ja, jeg kan åbne den i min browser og den ser ud til at være gemt

Perfekt! Nu er vi klar til at bygge vores app. Lad os starte med at rense op i src mappen og opsætte vores første komponent.

1. Først, i src mappen, åbn App.jsx og erstat alt indholdet med denne simple startstruktur:

jsx

Copy

```
function App() {  
  return (  
    <div className="min-h-screen bg-gray-50 p-4">  
      <h1 className="text-2xl font-bold text-center">Trænings Tracker</h1>  
    </div>  
  )  
}
```

export default App

2. Du kan også slette disse filer fra src mappen, da vi ikke skal bruge dem:

- App.css
- assets mappen

Efter du har gjort dette, burde du se en simpel overskrift centreret på en lysegrå baggrund i din browser.

Fortæl mig når du har gjort dette, så går vi videre med at bygge vores første rigtige komponent for træningsregistrering.

CopyRetry

## EW

Det er gjort

Super! Nu skal vi lave vores første rigtige komponent til at registrere træninger. Lad os oprette en ny mappe til vores komponenter og lave vores første komponent.

1. Opret en ny mappe kaldet components i src mappen
2. I components mappen, opret en ny fil kaldet WorkoutForm.jsx

I WorkoutForm.jsx, indsæt denne startkode:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus } from 'lucide-react'
```

```
function WorkoutForm() {  
  const [exercise, setExercise] = useState({  
    name: "",  
    sets: "",  
    reps: "",  
    weight: ""  
  })  
  
  const handleSubmit = (e) => {  
    e.preventDefault()  
    // TODO: Håndter gemning af øvelse  
    console.log('Exercise submitted:', exercise)  
  }  
  
  return (  
    <div className="max-w-md mx-auto bg-white rounded-lg shadow p-6 mt-4">  
      <h2 className="text-xl font-semibold mb-4">Tilføj øvelse</h2>
```

```
<form onSubmit={handleSubmit} className="space-y-4">

  <div>

    <input

      type="text"

      placeholder="Øvelsesnavn"

      value={exercise.name}

      onChange={(e) => setExercise({...exercise, name: e.target.value})}

      className="w-full p-2 border rounded"

    />

  </div>

  <div className="grid grid-cols-3 gap-2">

    <input

      type="number"

      placeholder="Sæt"

      value={exercise.sets}

      onChange={(e) => setExercise({...exercise, sets: e.target.value})}

      className="p-2 border rounded"

    />

    <input

      type="number"

      placeholder="Reps"

      value={exercise.reps}

      onChange={(e) => setExercise({...exercise, reps: e.target.value})}

      className="p-2 border rounded"

    />

    <input

      type="number"
```

```

    placeholder="Vægt (kg)"
    value={exercise.weight}
    onChange={(e) => setExercise({...exercise, weight: e.target.value})}
    className="p-2 border rounded"
  />
</div>

<button
  type="submit"
  className="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
items-center justify-center gap-2"
  >
  <Plus size={20} />
  Tilføj øvelse
</button>
</form>
</div>
)
}

```

export default WorkoutForm

3. Opdater App.jsx til at bruge den nye komponent:

jsx

Copy

```
import WorkoutForm from './components/WorkoutForm'
```

```
function App() {
```

```
  return (
```

```
<div className="min-h-screen bg-gray-50 p-4">

  <h1 className="text-2xl font-bold text-center">Trænings Tracker</h1>

  <WorkoutForm />

</div>

)

}
```

export default App

Dette vil give os en formular hvor vi kan indtaste øvelser. Fortæl mig når du har oprettet disse filer, så kan vi teste at det virker og tilføje mere funktionalitet.

CopyRetry

**EW**

Hvordan opretter jeg mapper under src? Højreklik` ?

Ja, du kan højreklikke på src mappen i VS Code's filbrowser og vælge "New Folder". Alternativt kan du:

1. Højreklik på src mappen
2. Vælg "New Folder"
3. Skriv "components"

Efter du har oprettet mappen, kan du oprette WorkoutForm.jsx filen ved at:

1. Højreklik på den nye components mappe
2. Vælg "New File"
3. Skriv "WorkoutForm.jsx"

Skal jeg guide dig gennem processen trin for trin?

CopyRetry

**EW**

Nu har jeg fuldført de to trin

Perfekt! Lad os se om det virker. Tjek i din browser på <http://localhost:5173> - du burde nu se:

1. En overskrift "Trænings Tracker"
2. En formular med:
  - Et input felt til øvelsesnavn
  - Tre numeriske input felter for sæt, reps og vægt
  - En blå "Tilføj øvelse" knap med et plus-ikon

Hvis du udfylder formularen og trykker på knappen, burde du kunne se data'en blive logget i browserens konsol (højreklik -> Inspect -> Console).

Virker det som forventet? Hvis ja, kan vi gå videre med at tilføje funktionalitet til at:

1. Gemme øvelserne i en liste
2. Vise de gemte øvelser
3. Tilføje mulighed for at slette øvelser

CopyRetry

Claude does not have internet access. Links provided may not be accurate or up to date.

## EW

Jep, det ser sådan her ud

Perfekt! Nu er den grundlæggende struktur på plads. Lad os tilføje funktionalitet til at gemme og vise øvelser.

1. Først skal vi opdatere App.jsx til at håndtere listen af øvelser:

jsx

Copy

```
import { useState } from 'react'
```

```
import WorkoutForm from './components/WorkoutForm'

function App() {
  const [exercises, setExercises] = useState([])

  const addExercise = (exercise) => {
    setExercises([...exercises, { ...exercise, id: Date.now() }])
  }

  const deleteExercise = (id) => {
    setExercises(exercises.filter(exercise => exercise.id !== id))
  }

  return (
    <div className="min-h-screen bg-gray-50 p-4">
      <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>
      <WorkoutForm onSubmit={addExercise} />

      {exercises.length > 0 && (
        <div className="max-w-md mx-auto mt-8">
          <h2 className="text-xl font-semibold mb-4">Dine øvelser</h2>
          <div className="space-y-2">
            {exercises.map(exercise => (
              <div
                key={exercise.id}
                className="bg-white p-4 rounded-lg shadow flex justify-between items-center"
              >
```



```

<div>

  <h3 className="font-medium">{exercise.name}</h3>

  <p className="text-gray-600">

    {exercise.sets} sæt × {exercise.reps} reps

    {exercise.weight} & ` @ ${exercise.weight}kg `

  </p>

</div>

<button

  onClick={() => deleteExercise(exercise.id)}

  className="text-red-500 hover:text-red-700"

>

  Slet

</button>

</div>

  )}

</div>

</div>

)}

</div>

)

}

```

export default App

2. Og opdater WorkoutForm.jsx til at bruge den nye onSubmit prop:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus } from 'lucide-react'
```

```
function WorkoutForm({ onSubmit }) {
```

```
  const [exercise, setExercise] = useState({
```

```
    name: "",
```

```
    sets: "",
```

```
    reps: "",
```

```
    weight: ""
```

```
  })
```

```
  const handleSubmit = (e) => {
```

```
    e.preventDefault()
```

```
    if (!exercise.name || !exercise.sets || !exercise.reps) return
```

```
    onSubmit(exercise)
```

```
    // Nulstil formen
```

```
    setExercise({
```

```
      name: "",
```

```
      sets: "",
```

```
      reps: "",
```

```
      weight: ""
```

```
    })
```

```
  }
```

```
  return (
```

```
    <div className="max-w-md mx-auto bg-white rounded-lg shadow p-6">
```

```
<h2 className="text-xl font-semibold mb-4">Tilføj øvelse</h2>

<form onSubmit={handleSubmit} className="space-y-4">

  <div>

    <input

      type="text"

      placeholder="Øvelsesnavn"

      value={exercise.name}

      onChange={(e) => setExercise({...exercise, name: e.target.value})}

      className="w-full p-2 border rounded"

      required

    />

  </div>

  <div className="grid grid-cols-3 gap-2">

    <input

      type="number"

      placeholder="Sæt"

      value={exercise.sets}

      onChange={(e) => setExercise({...exercise, sets: e.target.value})}

      className="p-2 border rounded"

      required

      min="1"

    />

    <input

      type="number"

      placeholder="Reps"

      value={exercise.reps}

      onChange={(e) => setExercise({...exercise, reps: e.target.value})}
```

```

      className="p-2 border rounded"
      required
      min="1"
    />
    <input
      type="number"
      placeholder="Vægt (kg)"
      value={exercise.weight}
      onChange={(e) => setExercise({...exercise, weight: e.target.value})}
      className="p-2 border rounded"
      min="0"
    />
  </div>
  <button
    type="submit"
    className="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
items-center justify-center gap-2"
  >
    <Plus size={20} />
    Tilføj øvelse
  </button>
</form>
</div>
)
}

export default WorkoutForm

```

Nu skulle du kunne:

1. Tilføje øvelser ved at udfylde formularen
2. Se en liste over tilføjede øvelser under formularen
3. Slette øvelser ved at klikke på "Slet" knappen

Prøv det og fortæl mig hvis det virker som forventet. Derefter kan vi tilføje:

1. Lokalt storage så øvelserne gemmes mellem sessioner
2. Mulighed for at redigere eksisterende øvelser
3. Gruppering af øvelser i træningsprogrammer

CopyRetry

## EW

Her er et par prøveindtastninger

Perfekt! Systemet virker som det skal. Lad os nu tilføje lokal lagring så øvelserne bliver gemt selv når du lukker browseren.

1. Først skal vi opdatere App.jsx til at bruge localStorage:

jsx

Copy

```
import { useState, useEffect } from 'react'
```

```
import WorkoutForm from './components/WorkoutForm'
```

```
function App() {
```

```
  const [exercises, setExercises] = useState(() => {
```

```
// Indlæs gemte øvelser fra localStorage når appen starter
const savedExercises = localStorage.getItem('exercises')
return savedExercises ? JSON.parse(savedExercises) : []
})
```

```
// Gem øvelser i localStorage hver gang de opdateres
useEffect(() => {
  localStorage.setItem('exercises', JSON.stringify(exercises))
}, [exercises])
```

```
const addExercise = (exercise) => {
  setExercises([...exercises, { ...exercise, id: Date.now() }])
}
```

```
const deleteExercise = (id) => {
  setExercises(exercises.filter(exercise => exercise.id !== id))
}
```

```
return (
  <div className="min-h-screen bg-gray-50 p-4">
    <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>
    <WorkoutForm onSubmit={addExercise} />

    {exercises.length > 0 && (
      <div className="max-w-md mx-auto mt-8">
        <h2 className="text-xl font-semibold mb-4">Dine øvelser</h2>
        <div className="space-y-2">
```

```

{exercises.map(exercise => (
  <div
    key={exercise.id}
    className="bg-white p-4 rounded-lg shadow flex justify-between items-center"
  >
    <div>
      <h3 className="font-medium">{exercise.name}</h3>
      <p className="text-gray-600">
        {exercise.sets} sæt × {exercise.reps} reps
        {exercise.weight && ` @ ${exercise.weight}kg`}
      </p>
    </div>
    <button
      onClick={() => deleteExercise(exercise.id)}
      className="px-3 py-1 text-sm text-red-500 hover:text-red-700 hover:bg-red-50
rounded"
    >
      Slet
    </button>
  </div>
  )})
</div>
</div>
)
}

```

export default App

Nu vil øvelserne blive gemt lokalt i browseren. Prøv at:

1. Tilføje nogle øvelser
2. Genindlæs siden (F5) - øvelserne burde stadig være der
3. Luk browseren og åbn siden igen - øvelserne burde stadig være der

Skal vi derefter tilføje:

1. Mulighed for at redigere eksisterende øvelser?
2. Gruppering af øvelser i træningsprogrammer?
3. Eller noget helt tredje?

CopyRetry

**EW**

Ja, nu må du gerne hjælpe mig med at opbygge webapplikationen, så brugeren kan 1) opbygge sit træningsprogram/vælge de relevante øvelser og gentagelser. 2) gøre det muligt for brugeren at vælge træningsprogrammet til en konkret træning, 3) indtaste de opnåede gentagelser under hver øvelse (eksempelvis 8 gentagelser på første runde pullups, herefter 6, herefter 5 og til sidst 5). til sidst skal brugeren have mulighed for at gemme og eksportere træningen i et passende filformat. Brugeren vil typisk køre programmet via en browser på mobilen.

God plan! Lad os omstrukturere applikationen til at håndtere denne workflow. Vi starter med at lave de nødvendige komponenter.

1. Først skal vi oprette nogle nye komponenter. I din components mappe, opret følgende nye filer:

jsx

Copy

*// components/ProgramBuilder.jsx - For at oprette træningsprogrammer*

```
import { useState } from 'react'
```

```
import { Plus, Save } from 'lucide-react'
```

```
function ProgramBuilder({ onSaveProgram }) {
```



```

const [programName, setProgramName] = useState("")
const [exercises, setExercises] = useState([])

const addExercise = (exercise) => {
  setExercises([...exercises, {
    ...exercise,
    id: Date.now(),
    plannedSets: Number(exercise.sets),
    actualSets: [] // Her gemmer vi de faktiske sæt når træningen udføres
  ]])
}

const saveProgram = () => {
  if (!programName.trim()) return
  onSaveProgram({
    id: Date.now(),
    name: programName,
    exercises: exercises,
    createdAt: new Date().toISOString()
  })
  setProgramName("")
  setExercises([])
}

return (
  <div className="bg-white rounded-lg shadow p-4">
    <h2 className="text-xl font-semibold mb-4">Opret træningsprogram</h2>

```

```
<input
  type="text"
  placeholder="Programnavn"
  value={programName}
  onChange={(e) => setProgramName(e.target.value)}
  className="w-full p-2 border rounded mb-4"
/>
```

```
<WorkoutForm onSubmit={addExercise} />
```

```
<div className="mt-4 space-y-2">
  {exercises.map(exercise => (
    <div key={exercise.id} className="border p-2 rounded">
      <p className="font-medium">{exercise.name}</p>
      <p className="text-sm text-gray-600">
        {exercise.plannedSets} sæt × {exercise.reps} reps
        {exercise.weight && ` @ ${exercise.weight}kg` }
      </p>
    </div>
  ))}
</div>
```

```
{exercises.length > 0 && (
  <button
    onClick={saveProgram}
    className="mt-4 w-full bg-green-500 text-white py-2 px-4 rounded hover:bg-green-600
flex items-center justify-center gap-2"
  >
```

```

      <Save size={20} />
      Gem program
    </button>
  )}
</div>
)
}

```

```
export default ProgramBuilder
```

```
jsx
```

```
Copy
```

```
// components/ActiveWorkout.jsx - For at registrere en aktiv træning
```

```
import { useState } from 'react'
```

```
import { Check, Save } from 'lucide-react'
```

```

function ActiveWorkout({ program, onComplete }) {
  const [currentExercises, setCurrentExercises] = useState(
    program.exercises.map(ex => ({
      ...ex,
      actualSets: []
    }))
  )

```

```

  const addSet = (exercisId, reps) => {
    setCurrentExercises(exercises =>
      exercises.map(ex =>
        ex.id === exercisId

```

```

      ? { ...ex, actualSets: [...ex.actualSets, Number(reps)] }
      : ex
    )
  )
}

```

```

const handleComplete = () => {
  onComplete({
    programId: program.id,
    programName: program.name,
    date: new Date().toISOString(),
    exercises: currentExercises,
  })
}

```

```

return (
  <div className="bg-white rounded-lg shadow p-4">
    <h2 className="text-xl font-semibold mb-4">
      {program.name} - Aktiv træning
    </h2>

    <div className="space-y-6">
      {currentExercises.map(exercise => (
        <div key={exercise.id} className="border-b pb-4">
          <h3 className="font-medium mb-2">{exercise.name}</h3>
          <p className="text-sm text-gray-600 mb-2">
            Mål: {exercise.plannedSets} sæt × {exercise.reps} reps
          </p>
        </div>
      ))}
    </div>
  </div>
)

```

```
{exercise.weight} @ ${exercise.weight}kg` }
```

```
</p>
```

```
<div className="space-y-2">
```

```
{Array.from({ length: exercise.plannedSets }).map((_, idx) => (
```

```
<div key={idx} className="flex gap-2">
```

```
<input
```

```
type="number"
```

```
placeholder={` Sæt ${idx + 1} reps` }
```

```
value={exercise.actualSets[idx] || ""}
```

```
onChange={(e) => addSet(exercise.id, e.target.value)}
```

```
className="w-24 p-2 border rounded"
```

```
min="0"
```

```
/>
```

```
{exercise.actualSets[idx] && (
```

```
<Check className="text-green-500" size={24} />
```

```
)}
```

```
</div>
```

```
)))
```

```
</div>
```

```
</div>
```

```
)))
```

```
</div>
```

```
<button
```

```
onClick={handleComplete}
```

```
      className="mt-6 w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
      items-center justify-center gap-2"
```

```
>
```

```
<Save size={20} />
```

```
Afslut træning
```

```
</button>
```

```
</div>
```

```
)
```

```
}
```

```
export default ActiveWorkout
```

2. Opdater App.jsx til at håndtere denne nye struktur:

```
jsx
```

```
Copy
```

```
import { useState, useEffect } from 'react'
```

```
import ProgramBuilder from './components/ProgramBuilder'
```

```
import ActiveWorkout from './components/ActiveWorkout'
```

```
function App() {
```

```
  const [programs, setPrograms] = useState(() => {
```

```
    const saved = localStorage.getItem('programs')
```

```
    return saved ? JSON.parse(saved) : []
```

```
  })
```

```
  const [workoutHistory, setWorkoutHistory] = useState(() => {
```

```
    const saved = localStorage.getItem('workoutHistory')
```

```
    return saved ? JSON.parse(saved) : []
```

```
}}
```

```
const [activeProgram, setActiveProgram] = useState(null)
```

```
// Gem data i localStorage
```

```
useEffect(() => {
```

```
  localStorage.setItem('programs', JSON.stringify(programs))
```

```
  localStorage.setItem('workoutHistory', JSON.stringify(workoutHistory))
```

```
}, [programs, workoutHistory])
```

```
const handleSaveProgram = (program) => {
```

```
  setPrograms([...programs, program])
```

```
}
```

```
const handleCompleteWorkout = (workout) => {
```

```
  setWorkoutHistory([...workoutHistory, workout])
```

```
  setActiveProgram(null)
```

```
}
```

```
const exportWorkoutHistory = () => {
```

```
  const data = JSON.stringify(workoutHistory, null, 2)
```

```
  const blob = new Blob([data], { type: 'application/json' })
```

```
  const url = URL.createObjectURL(blob)
```

```
  const a = document.createElement('a')
```

```
  a.href = url
```

```
  a.download = 'workout-history.json'
```

```
  document.body.appendChild(a)
```

```

a.click()

document.body.removeChild(a)

URL.revokeObjectURL(url)
}

return (
  <div className="min-h-screen bg-gray-50 p-4">
    <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>

    {!activeProgram ? (
      <div className="space-y-6">
        <ProgramBuilder onSaveProgram={handleSaveProgram} />

        {programs.length > 0 && (
          <div className="bg-white rounded-lg shadow p-4">
            <h2 className="text-xl font-semibold mb-4">Dine programmer</h2>
            <div className="space-y-2">
              {programs.map(program => (
                <button
                  key={program.id}
                  onClick={() => setActiveProgram(program)}
                  className="w-full text-left p-3 border rounded hover:bg-gray-50"
                >
                  <p className="font-medium">{program.name}</p>
                  <p className="text-sm text-gray-600">
                    {program.exercises.length} øvelser
                  </p>
                </button>
              )}
            </div>
          </div>
        )}
      </div>
    )}
  </div>
)

```



```
    </button>
```

```
  )}
```

```
</div>
```

```
</div>
```

```
}}
```

```
{workoutHistory.length > 0 && (
```

```
  <button
```

```
    onClick={exportWorkoutHistory}
```

```
    className="w-full bg-gray-500 text-white py-2 px-4 rounded hover:bg-gray-600"
```

```
  >
```

```
    Eksporter træningshistorik
```

```
  </button>
```

```
}}
```

```
</div>
```

```
): (
```

```
  <ActiveWorkout
```

```
    program={activeProgram}
```

```
    onComplete={handleCompleteWorkout}
```

```
  />
```

```
}}
```

```
</div>
```

```
)
```

```
}
```

```
export default App
```

Denne opdatering giver os:

1. Mulighed for at oprette træningsprogrammer med øvelser
2. Vælge et program og starte en træning
3. Registrere faktiske sæt og gentagelser under træningen
4. Gemme træningshistorik og eksportere den

Vil du have mig til at forklare nogle specifikke dele af koden eller skal vi implementere flere features?

CopyRetry

**EW**

Skal jeg slette filen workoutform?

Nej, behold WorkoutForm.jsx - vi skal stadig bruge den som en del af ProgramBuilder komponenten. Vi skal dog opdatere den lidt for at passe bedre ind i den nye struktur.

Lad os opdatere WorkoutForm.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus } from 'lucide-react'
```

```
function WorkoutForm({ onSubmit }) {  
  const [exercise, setExercise] = useState({  
    name: "",  
    sets: "",  
    reps: "",  
    weight: ""  
  })
```

```
  const handleSubmit = (e) => {  
    e.preventDefault()
```

```
if (!exercise.name || !exercise.sets || !exercise.reps) return
```

```
onSubmit(exercise)
```

```
// Nulstil formen
```

```
setExercise({
```

```
  name: "",
```

```
  sets: "",
```

```
  reps: "",
```

```
  weight: ""
```

```
})
```

```
}
```

```
return (
```

```
<div className="bg-white rounded-lg">
```

```
<form onSubmit={handleSubmit} className="space-y-4">
```

```
<div>
```

```
<input
```

```
  type="text"
```

```
  placeholder="Øvelsesnavn"
```

```
  value={exercise.name}
```

```
  onChange={(e) => setExercise({...exercise, name: e.target.value})}
```

```
  className="w-full p-2 border rounded"
```

```
  required
```

```
/>
```

```
</div>
```

```
<div className="grid grid-cols-3 gap-2">
```

```
<input
  type="number"
  placeholder="Sæt"
  value={exercise.sets}
  onChange={(e) => setExercise({...exercise, sets: e.target.value})}
  className="p-2 border rounded"
  required
  min="1"
/>
```

```
<input
  type="number"
  placeholder="Reps"
  value={exercise.reps}
  onChange={(e) => setExercise({...exercise, reps: e.target.value})}
  className="p-2 border rounded"
  required
  min="1"
/>
```

```
<input
  type="number"
  placeholder="Vægt (kg)"
  value={exercise.weight}
  onChange={(e) => setExercise({...exercise, weight: e.target.value})}
  className="p-2 border rounded"
  min="0"
/>
```

```
</div>
```

```

<button
  type="submit"
  className="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
items-center justify-center gap-2"
  >
    <Plus size={20} />
    Tilføj øvelse
  </button>
</form>
</div>
)
}

```

export default WorkoutForm

Den opdaterede version fjerner den ekstra styling og overskrift, da den nu vil blive brugt som en underkomponent i ProgramBuilder.

Fortæl mig når du har opdateret WorkoutForm.jsx, så kan vi teste at alt virker sammen.

CopyRetry

**20.37.45 [vite] Internal server error: C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by Mangaard\Skrivebord\Kodning\Workout Tracker\workout-tracker\src\components\WorkoutForm.jsx: Unexpecte**

**pasted**

**EW**

Fik den her besked i terminalen

Fejlen ser ud til at være i min="1" attributten. I React skal vi bruge curly braces for numeriske værdier. Lad os opdatere WorkoutForm.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus } from 'lucide-react'
```

```
function WorkoutForm({ onSubmit }) {
```

```
  const [exercise, setExercise] = useState({
```

```
    name: "",
```

```
    sets: "",
```

```
    reps: "",
```

```
    weight: ""
```

```
  })
```

```
  const handleSubmit = (e) => {
```

```
    e.preventDefault()
```

```
    if (!exercise.name || !exercise.sets || !exercise.reps) return
```

```
    onSubmit(exercise)
```

```
    // Nulstil formen
```

```
    setExercise({
```

```
      name: "",
```

```
      sets: "",
```

```
      reps: "",
```

```
      weight: ""
```

```
    })
```

```
  }
```

```
  return (
```

```
<div className="bg-white rounded-lg">
  <form onSubmit={handleSubmit} className="space-y-4">
    <div>
      <input
        type="text"
        placeholder="Øvelsesnavn"
        value={exercise.name}
        onChange={(e) => setExercise({...exercise, name: e.target.value})}
        className="w-full p-2 border rounded"
        required
      />
    </div>
    <div className="grid grid-cols-3 gap-2">
      <input
        type="number"
        placeholder="Sæt"
        value={exercise.sets}
        onChange={(e) => setExercise({...exercise, sets: e.target.value})}
        className="p-2 border rounded"
        required
        min={1}
      />
      <input
        type="number"
        placeholder="Reps"
        value={exercise.reps}
        onChange={(e) => setExercise({...exercise, reps: e.target.value})}
```

```

      className="p-2 border rounded"
      required
      min={1}
    />
    <input
      type="number"
      placeholder="Vægt (kg)"
      value={exercise.weight}
      onChange={(e) => setExercise({...exercise, weight: e.target.value})}
      className="p-2 border rounded"
      min={0}
    />
  </div>
  <button
    type="submit"
    className="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
items-center justify-center gap-2"
  >
    <Plus size={20} />
    Tilføj øvelse
  </button>
</form>
</div>
)
}

export default WorkoutForm

```



De vigtigste ændringer er:

- `min="1"` er ændret til `min={1}`
- `min="0"` er ændret til `min={0}`

Dette er fordi React forventer JavaScript værdier i curly braces i stedet for string værdier i anførselstegn for numeriske attributter.

Prøv at gemme denne version og se om fejlen forsvinder.

CopyRetry

**20.38.38 [vite] (client) hmr update /src/components/WorkoutForm.jsx 20.38.38 [vite]**  
**Internal server error: C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by**  
**Mangaard\Skrivebord\Kodning\Work**

**pasted**

**EW**

Får den her besked

Lad os forenkle WorkoutForm.jsx ved at fjerne min-attributterne helt, da de måske skaber problemer. Her er en simplere version:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus } from 'lucide-react'
```

```
function WorkoutForm({ onSubmit }) {  
  const [exercise, setExercise] = useState({  
    name: "",  
    sets: "",  
    reps: "",  
    weight: ""  
  })
```

```

const handleSubmit = (e) => {
  e.preventDefault()
  if (!exercise.name || !exercise.sets || !exercise.reps) return

  onSubmit(exercise)

  // Nulstil formen
  setExercise({
    name: "",
    sets: "",
    reps: "",
    weight: ""
  })
}

return (
  <div className="bg-white rounded-lg">
    <form onSubmit={handleSubmit} className="space-y-4">
      <div>
        <input
          type="text"
          placeholder="Øvelsesnavn"
          value={exercise.name}
          onChange={(e) => setExercise({...exercise, name: e.target.value})}
          className="w-full p-2 border rounded"
          required
        />

```

```
</div>
```

```
<div className="grid grid-cols-3 gap-2">
```

```
  <input
```

```
    type="number"
```

```
    placeholder="Sæt"
```

```
    value={exercise.sets}
```

```
    onChange={(e) => setExercise({...exercise, sets: e.target.value})}
```

```
    className="p-2 border rounded"
```

```
    required
```

```
  <input
```

```
    type="number"
```

```
    placeholder="Reps"
```

```
    value={exercise.reps}
```

```
    onChange={(e) => setExercise({...exercise, reps: e.target.value})}
```

```
    className="p-2 border rounded"
```

```
    required
```

```
  <input
```

```
    type="number"
```

```
    placeholder="Vægt (kg)"
```

```
    value={exercise.weight}
```

```
    onChange={(e) => setExercise({...exercise, weight: e.target.value})}
```

```
    className="p-2 border rounded"
```

```
</div>
```

```
<button
```

```

      type="submit"

      className="w-full bg-blue-500 text-white py-2 px-4 rounded hover:bg-blue-600 flex
items-center justify-center gap-2"

      >

      <Plus size={20} />

      Tilføj øvelse

    </button>

  </form>

</div>

)

}

```

export default WorkoutForm

Prøv at erstatte indholdet i din WorkoutForm.jsx med denne kode og se om det virker. Vi kan altid tilføje validering senere på en anden måde.

CopyRetry

## EW

Det virkede, men kan ikke se noget i localhost længere

Det er fordi vi også skal oprette og konfigurere ProgramBuilder komponenten og opdatere App.jsx. Lad os starte med App.jsx:

1. Opdater App.jsx med denne kode:

jsx

Copy

```
import { useState } from 'react'
```

```
import WorkoutForm from './components/WorkoutForm'
```

```
function App() {
```

```
  const [exercises, setExercises] = useState([])
```

```
const addExercise = (exercise) => {  
  setExercises([...exercises, { ...exercise, id: Date.now() }])  
}
```

```
return (  
  <div className="min-h-screen bg-gray-50 p-4">  
    <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>  
    <WorkoutForm onSubmit={addExercise} />  
  
    {exercises.length > 0 && (  
      <div className="max-w-md mx-auto mt-8">  
        <h2 className="text-xl font-semibold mb-4">Dine øvelser</h2>  
        <div className="space-y-2">  
          {exercises.map(exercise => (  
            <div  
              key={exercise.id}  
              className="bg-white p-4 rounded-lg shadow flex justify-between items-center"  
            >  
              <div>  
                <h3 className="font-medium">{exercise.name}</h3>  
                <p className="text-gray-600">  
                  {exercise.sets} sæt × {exercise.reps} reps  
                  {exercise.weight && ` @ ${exercise.weight}kg` }  
                </p>  
              </div>  
            </div>  
          )}  
        </div>  
      )  
    )  
  )  
)
```

```
    )})  
  </div>  
</div>  
}  
</div>  
)  
}
```

export default App

Dette burde gendanne den grundlæggende funktionalitet, hvor du kan:

1. Se overskriften
2. Tilføje øvelser via formularen
3. Se de tilføjede øvelser i en liste nedenunder

Fortæl mig når du kan se appen igen, så kan vi fortsætte med at tilføje ProgramBuilder og de andre komponenter.

CopyRetry

## **EW**

Nu er jeg her, men det er jo samme struktur, som sidst

Ja, du har ret. Lad os implementere den nye struktur hvor brugeren kan:

1. Oprette træningsprogrammer
2. Vælge et program til træning

### 3. Registrere faktiske reps under træningen

Først skal vi oprette en ny komponent ProgramBuilder.jsx i components mappen:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Plus, Save } from 'lucide-react'
```

```
import WorkoutForm from './WorkoutForm'
```

```
function ProgramBuilder() {
```

```
  const [programName, setProgramName] = useState("")
```

```
  const [exercises, setExercises] = useState([])
```

```
  const addExercise = (exercise) => {
```

```
    setExercises([...exercises, { ...exercise, id: Date.now() }])
```

```
}
```

```
const saveProgram = () => {
```

```
  const program = {
```

```
    id: Date.now(),
```

```
    name: programName,
```

```
    exercises: exercises,
```

```
    createdAt: new Date().toISOString()
```

```
}
```

```
// Gem til localStorage
```

```
const existingPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')
```

```
localStorage.setItem('workoutPrograms', JSON.stringify([...existingPrograms, program]))
```

```
// Nulstil form
```

```
setProgramName("")
```

```
setExercises([])
```

```
}
```

```
return (
```

```
<div className="max-w-md mx-auto bg-white rounded-lg shadow p-6">
```

```
<h2 className="text-xl font-bold mb-4">Opret nyt træningsprogram</h2>
```

```
<input
```

```
  type="text"
```

```
  placeholder="Programnavn"
```

```
  value={programName}
```

```
  onChange={(e) => setProgramName(e.target.value)}
```

```
  className="w-full p-2 border rounded mb-4"
```

```
<WorkoutForm onSubmit={addExercise} />
```

```
<div className="mt-6">
```

```
<h3 className="font-semibold mb-2">Øvelser i programmet:</h3>
```

```
<div className="space-y-2">
```

```
  {exercises.map(exercise => (
```

```
    <div key={exercise.id} className="bg-gray-50 p-3 rounded">
```

```
      <p className="font-medium">{exercise.name}</p>
```

```
      <p className="text-sm text-gray-600">
```



```

        {exercise.sets} sæt × {exercise.reps} reps
        {exercise.weight ? ` @ ${exercise.weight}kg` : ""}
    </p>
</div>
)}}
</div>
</div>

```

```

{exercises.length > 0 && (
    <button
        onClick={saveProgram}
        className="mt-4 w-full bg-green-500 text-white py-2 px-4 rounded hover:bg-green-600
flex items-center justify-center gap-2"
        disabled={!programName.trim()}
    >
        <Save size={20} />
        Gem program
    </button>
)}
</div>
)
}

```

export default ProgramBuilder

Og opdater App.jsx:

jsx

Copy

Forklare

```
import { useState, useEffect } from 'react'
```

```
import ProgramBuilder from './components/ProgramBuilder'
```

```
function App() {
```

```
  const [view, setView] = useState('programs') // 'programs' eller 'create'
```

```
  const [programs, setPrograms] = useState([])
```

```
  // Indlæs gemte programmer
```

```
  useEffect(() => {
```

```
    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms')) || '[]')
```

```
    setPrograms(savedPrograms)
```

```
  }, [])
```

```
  return (
```

```
    <div className="min-h-screen bg-gray-50 p-4">
```

```
      <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>
```

```
      <div className="max-w-md mx-auto mb-6">
```

```
        <div className="flex gap-2">
```

```
          <button
```

```
            onClick={() => setView('programs')}>
```

```
            className={`flex-1 py-2 px-4 rounded ${
```

```
              view === 'programs'
```

```
                ? 'bg-blue-500 text-white'
```

```
                : 'bg-gray-200 text-gray-700'
```

```
            }`>
```

>

Mine programmer

</button>

<button

onClick={() => setView('create')}

className={`flex-1 py-2 px-4 rounded \${

view === 'create'

? 'bg-blue-500 text-white'

: 'bg-gray-200 text-gray-700'

}`}

>

Opret nyt program

</button>

</div>

</div>

{view === 'create' ? (

<ProgramBuilder />

): (

<div className="max-w-md mx-auto">

<div className="space-y-4">

{programs.map(program => (

<div key={program.id} className="bg-white p-4 rounded-lg shadow">

<h3 className="text-lg font-semibold">{program.name}</h3>

<p className="text-sm text-gray-600 mb-2">

{program.exercises.length} øvelser

</p>

```

        <button className="text-blue-500 text-sm hover:text-blue-700">
            Start træning
        </button>
    </div>
    )}]
</div>
</div>
)}
</div>
)
}

```

export default App

Dette giver os:

1. En fane-baseret navigation mellem "Mine programmer" og "Opret nyt program"
2. Mulighed for at oprette programmer med navn og øvelser
3. Gemme programmer i localStorage
4. Se liste over gemte programmer

Skal vi teste dette først, og så tilføje funktionaliteten til at starte en træning og registrere gentagelser?

CopyRetry

## EW

Nu ser det sådan her ud

2 / 2

Perfekt! Nu har vi den grundlæggende struktur på plads. Lad os tilføje funktionaliteten for at starte en træning.

Først skal vi oprette en ny komponent `ActiveWorkout.jsx` i `components` mappen:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Save } from 'lucide-react'
```

```
function ActiveWorkout({ program, onComplete, onCancel }) {  
  const [exerciseResults, setExerciseResults] = useState(  
    program.exercises.map(exercise => ({  
      ...exercise,  
      setResults: Array(Number(exercise.sets)).fill("")  
    }))  
  )  
}
```

```
const updateSetResult = (exerciseIndex, setIndex, reps) => {  
  setExerciseResults(prev => {
```

```

const newResults = [...prev]
newResults[exerciseIndex] = {
  ...newResults[exerciseIndex],
  setResults: newResults[exerciseIndex].setResults.map((set, idx) =>
    idx === setIndex ? reps : set
  )
}
return newResults
})
}

```

```

const handleComplete = () => {
  const workoutResult = {
    programId: program.id,
    programName: program.name,
    date: new Date().toISOString(),
    exercises: exerciseResults,
  }
  onComplete(workoutResult)
}

```

```

return (
  <div className="max-w-md mx-auto bg-white rounded-lg shadow p-6">
    <h2 className="text-xl font-bold mb-6">{program.name}</h2>

    {exerciseResults.map((exercise, exerciseIndex) => (
      <div key={exercise.id} className="mb-8">

```

```
<h3 className="font-semibold text-lg mb-2">{exercise.name}</h3>
```

```
<p className="text-sm text-gray-600 mb-3">
```

```
  Mål: {exercise.sets} sæt × {exercise.reps} reps
```

```
  {exercise.weight ? ` @ ${exercise.weight}kg` : ""}
```

```
</p>
```

```
<div className="space-y-2">
```

```
  {exercise.setResults.map((result, setIndex) => (
```

```
    <div key={setIndex} className="flex items-center gap-2">
```

```
      <span className="text-sm text-gray-600">Sæt {setIndex + 1}</span>
```

```
      <input
```

```
        type="number"
```

```
        value={result}
```

```
        onChange={(e) => updateSetResult(exerciseIndex, setIndex, e.target.value)}
```

```
        placeholder="Antal reps"
```

```
        className="w-20 p-2 border rounded"
```

```
      />
```

```
      <span className="text-sm text-gray-600">reps</span>
```

```
    </div>
```

```
  )}]
```

```
</div>
```

```
</div>
```

```
  )}]
```

```
<div className="flex gap-2 mt-6">
```

```
  <button
```

```
    onClick={handleComplete}
```

```
      className="flex-1 bg-green-500 text-white py-2 px-4 rounded hover:bg-green-600 flex
items-center justify-center gap-2"
```

```
>
```

```
<Save size={20} />
```

```
Gem træning
```

```
</button>
```

```
<button
```

```
  onClick={onCancel}
```

```
  className="flex-1 bg-gray-200 text-gray-700 py-2 px-4 rounded hover:bg-gray-300"
```

```
>
```

```
  Annuller
```

```
</button>
```

```
</div>
```

```
</div>
```

```
)
```

```
}
```

```
export default ActiveWorkout
```

Og opdater App.jsx til at håndtere den aktive træning:

```
jsx
```

Copy

```
import { useState, useEffect } from 'react'
```

```
import ProgramBuilder from './components/ProgramBuilder'
```

```
import ActiveWorkout from './components/ActiveWorkout'
```

```
function App() {
```

```
  const [view, setView] = useState('programs') // 'programs', 'create', eller 'workout'
```



```

const [programs, setPrograms] = useState([])

const [activeProgram, setActiveProgram] = useState(null)

const [workoutHistory, setWorkoutHistory] = useState([])

// Indlæs gemte data
useEffect(() => {
  const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')
  const savedHistory = JSON.parse(localStorage.getItem('workoutHistory') || '[]')
  setPrograms(savedPrograms)
  setWorkoutHistory(savedHistory)
}, [])

const startWorkout = (program) => {
  setActiveProgram(program)
  setView('workout')
}

const completeWorkout = (workoutResult) => {
  const newHistory = [...workoutHistory, workoutResult]
  setWorkoutHistory(newHistory)
  localStorage.setItem('workoutHistory', JSON.stringify(newHistory))
  setActiveProgram(null)
  setView('programs')
}

return (
  <div className="min-h-screen bg-gray-50 p-4">

```

```
<h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>
```

```
{view !== 'workout' && (
```

```
<div className="max-w-md mx-auto mb-6">
```

```
<div className="flex gap-2">
```

```
<button
```

```
onClick={() => setView('programs')}
```

```
className={`flex-1 py-2 px-4 rounded ${
```

```
view === 'programs'
```

```
? 'bg-blue-500 text-white'
```

```
: 'bg-gray-200 text-gray-700'
```

```
}`}
```

```
>
```

```
Mine programmer
```

```
</button>
```

```
<button
```

```
onClick={() => setView('create')}
```

```
className={`flex-1 py-2 px-4 rounded ${
```

```
view === 'create'
```

```
? 'bg-blue-500 text-white'
```

```
: 'bg-gray-200 text-gray-700'
```

```
}`}
```

```
>
```

```
Opret nyt program
```

```
</button>
```

```
</div>
```

```
</div>
```

```
)}
```

```
{view === 'create' ? (
```

```
  <ProgramBuilder />
```

```
) : view === 'workout' ? (
```

```
  <ActiveWorkout
```

```
    program={activeProgram}
```

```
    onComplete={completeWorkout}
```

```
    onCancel={() => {
```

```
      setActiveProgram(null)
```

```
      setView('programs')
```

```
    }}
```

```
  />
```

```
): (
```

```
  <div className="max-w-md mx-auto">
```

```
    <div className="space-y-4">
```

```
      {programs.map(program => (
```

```
        <div key={program.id} className="bg-white p-4 rounded-lg shadow">
```

```
          <h3 className="text-lg font-semibold">{program.name}</h3>
```

```
          <p className="text-sm text-gray-600 mb-2">
```

```
            {program.exercises.length} øvelser
```

```
          </p>
```

```
          <button
```

```
            onClick={() => startWorkout(program)}
```

```
            className="bg-blue-500 text-white py-1 px-3 rounded text-sm hover:bg-blue-600"
```

```
          >
```

```
            Start træning
```

```
        </button>
    </div>
    )}
</div>
</div>
)}
</div>
)
}
```

export default App

Nu kan du:

1. Oprette træningsprogrammer med navn og øvelser
2. Se listen over dine programmer
3. Starte en træning fra et program
4. Registrere antal reps for hvert sæt
5. Gemme træningsresultaterne

Vil du teste det og se om det virker som forventet?

CopyRetry

Claude does not have internet access. Links provided may not be accurate or up to date.

## EW

Yes. Det ser sådan her ud. Nu vil jeg gerne have mulighed for at gemme/eksportere til et relevant format (excel eller lign), så jeg kan analysere performance på tværs af træninger. Det vil også være rart med en funktion, så gennemførte træninger kunne afsluttes og gemmes i et arkiv (afsluttede træninger). Og er webapplikationen mobiloptimeret?

Lad os forbedre disse aspekter. Først skal vi tilføje en "Gem træning" knap og eksportfunktionalitet.

1. Først, lad os mobiloptimere ActiveWorkout.jsx:

jsx

Copy

*// Opdater styling i ActiveWorkout.jsx*

```
<div className="w-full max-w-md mx-auto bg-white rounded-lg shadow p-4 sm:p-6"> {/}*  
Mindre padding på mobil */}
```

```
<h2 className="text-xl font-bold mb-4 break-words">{program.name}</h2>
```

```
{exerciseResults.map((exercise, exerciseIndex) => (
```

```
<div key={exercise.id} className="mb-6">
```

```
<h3 className="font-semibold text-lg mb-2 break-words">{exercise.name}</h3>
```

```
<p className="text-sm text-gray-600 mb-3">
```

```
Mål: {exercise.sets} sæt × {exercise.reps} reps
```

```
{exercise.weight ? ` @ ${exercise.weight}kg` : ''}
```

```
</p>
```

```
<div className="space-y-3"> {/}* Større mellemrum for touch */}
```

```
{exercise.setResults.map((result, setIndex) => (
```

```
<div key={setIndex} className="flex items-center gap-3">
```

```
<span className="text-sm text-gray-600 w-16">Sæt {setIndex + 1}</span>
```

```
<input
```

```

    type="number"
    value={result}
    onChange={(e) => updateSetResult(exerciseIndex, setIndex, e.target.value)}
    placeholder="Reps"
    className="w-24 p-3 border rounded text-lg" {/* Større input og tekst */}
    pattern="[0-9]*" {/* Numerisk tastatur på mobil */}
  />
  <span className="text-sm text-gray-600">reps</span>
</div>
  )}
</div>
</div>
  )}

  <div className="fixed bottom-0 left-0 right-0 p-4 bg-white border-t shadow-lg sm:relative
sm:shadow-none sm:border-t-0">
  <div className="flex gap-2 max-w-md mx-auto">
    <button
      onClick={handleComplete}
      className="flex-1 bg-green-500 text-white py-3 px-4 rounded-lg text-lg font-medium
      hover:bg-green-600 flex items-center justify-center gap-2"
    >
      <Save size={24} />
      Gem træning
    </button>
    <button
      onClick={onCancel}

```

```
className="flex-1 bg-gray-200 text-gray-700 py-3 px-4 rounded-lg text-lg font-medium  
hover:bg-gray-300"
```

```
>
```

```
  Annuler
```

```
</button>
```

```
</div>
```

```
</div>
```

```
</div>
```

## 2. Tilføj eksportfunktionalitet til Excel format:

jsx

Copy

*// Ny fil: utils/exportToExcel.js*

```
export const exportWorkoutToExcel = (workout) => {
```

```
  // Format data for Excel
```

```
  let csvContent = "data:text/csv;charset=utf-8,";
```

```
  // Header
```

```
  csvContent += `Træningsprogram: ${workout.programName}\n`;
```

```
  csvContent += `Dato: ${new Date(workout.date).toLocaleDateString()}\n\n`;
```

```
  // Data for hver øvelse
```

```
  workout.exercises.forEach(exercise => {
```

```
    csvContent += `${exercise.name}\n`;
```

```
    csvContent += `Mål: ${exercise.sets} sæt × ${exercise.reps} reps`;
```

```
    if (exercise.weight) csvContent += ` @ ${exercise.weight}kg`;
```

```
    csvContent += '\n';
```

*// Resultater for hvert sæt*

```
csvContent += "Sæt,Reps udført\n";
exercise.setResults.forEach((reps, index) => {
  csvContent += `${index + 1},${reps}\n`;
});
csvContent += '\n';
});
```

*// Download filen*

```
const encodedUri = encodeURIComponent(csvContent);
const link = document.createElement("a");
link.setAttribute("href", encodedUri);
link.setAttribute("download", `træning_${workout.programName}_${new
Date(workout.date).toLocaleDateString()}.csv`);
document.body.appendChild(link);
link.click();
document.body.removeChild(link);
};
```

3. Tilføj arkiv visning til App.jsx:

jsx

Copy

*// I App.jsx, tilføj en ny view state*

```
const [view, setView] = useState('programs') // 'programs', 'create', 'workout', 'history'
```

*// Tilføj en ny knap i navigationen*

```
<div className="flex gap-2 flex-wrap">
```

```
<button
```



```
onClick={() => setView('programs')}  
className={` flex-1 py-2 px-4 rounded ${  
  view === 'programs' ? 'bg-blue-500 text-white' : 'bg-gray-200 text-gray-700'  
}`}` }  
>
```

Mine programmer

</button>

<button

```
onClick={() => setView('create')}  
className={` flex-1 py-2 px-4 rounded ${  
  view === 'create' ? 'bg-blue-500 text-white' : 'bg-gray-200 text-gray-700'  
}`}` }  
>
```

Opret nyt

</button>

<button

```
onClick={() => setView('history')}  
className={` flex-1 py-2 px-4 rounded ${  
  view === 'history' ? 'bg-blue-500 text-white' : 'bg-gray-200 text-gray-700'  
}`}` }  
>
```

Historik

</button>

</div>

*// Tilføj historik visning*

{view === 'history' && (

```

<div className="max-w-md mx-auto">

  <div className="space-y-4">

    {workoutHistory.map((workout, index) => (

      <div key={index} className="bg-white p-4 rounded-lg shadow">

        <div className="flex justify-between items-start mb-2">

          <div>

            <h3 className="text-lg font-semibold">{workout.programName}</h3>

            <p className="text-sm text-gray-600">

              {new Date(workout.date).toLocaleDateString()}

            </p>

          </div>

          <button

            onClick={() => exportWorkoutToExcel(workout)}

            className="text-blue-500 hover:text-blue-700"

          >

            Eksportér

          </button>

        </div>

        <div className="space-y-2">

          {workout.exercises.map((exercise, exIndex) => (

            <div key={exIndex} className="text-sm">

              <p className="font-medium">{exercise.name}</p>

              <p className="text-gray-600">

                Resultater: {exercise.setResults.join(', ')} reps

              </p>

            </div>

          ))}

        </div>

      )

    )}

```

```
    </div>

  </div>

  )}}

</div>

</div>

}}
```

Disse ændringer giver:

1. Bedre mobil-optimering med:
  - Større touch targets
  - Fastgjort "Gem" knap i bunden
  - Numerisk tastatur for rep-input
  - Responsivt layout
2. Export til CSV format der kan åbnes i Excel
3. Historik visning med mulighed for at se og eksportere tidligere træninger

Vil du have mig til at implementere flere forbedringer? For eksempel:

1. Grafer over progression
2. Mulighed for at sammenligne træninger
3. Bedre filtrering i historikken
4. Automatisk backup af data

CopyRetry

**13.07.44 [vite] (client) hmr update /src/components/ActiveWorkout.jsx 13.07.44 [vite]**  
**Internal server error: C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by**  
**Mangaard\Skrivebord\Kodning\Wo**

**pasted**

**EW**

Får den her besked, når jeg vil opdatere ActiveWorkout.jsx

Fejlen opstår på grund af kommentarerne i JSX. Lad os opdatere ActiveWorkout.jsx uden inline-kommentarer:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Save } from 'lucide-react'
```

```
function ActiveWorkout({ program, onComplete, onCancel }) {
```

```
  const [exerciseResults, setExerciseResults] = useState(
```

```
    program.exercises.map(exercise => ({
```

```
      ...exercise,
```

```
      setResults: Array(Number(exercise.sets)).fill("")
```

```
    }))
```

```
  )
```

```
  const updateSetResult = (exerciseIndex, setIndex, reps) => {
```

```
    setExerciseResults(prev => {
```

```
      const newResults = [...prev]
```

```
      newResults[exerciseIndex] = {
```

```
        ...newResults[exerciseIndex],
```

```
        setResults: newResults[exerciseIndex].setResults.map((set, idx) =>
```

```
          idx === setIndex ? reps : set
```

```
        )
```

```
      }
```

```
      return newResults
```

```
    })
```

```
  }
```

```
  const handleComplete = () => {
```

```

const workoutResult = {
  programId: program.id,
  programName: program.name,
  date: new Date().toISOString(),
  exercises: exerciseResults,
}

onComplete(workoutResult)
}

```

```

return (

```

```

  <div className="w-full max-w-md mx-auto bg-white rounded-lg shadow p-4 sm:p-6">
    <h2 className="text-xl font-bold mb-4 break-words">{program.name}</h2>

```

```

    {exerciseResults.map((exercise, exerciseIndex) => (

```

```

      <div key={exercise.id} className="mb-6">

```

```

        <h3 className="font-semibold text-lg mb-2 break-words">{exercise.name}</h3>

```

```

        <p className="text-sm text-gray-600 mb-3">

```

```

          Mål: {exercise.sets} sæt × {exercise.reps} reps

```

```

          {exercise.weight ? ` @ ${exercise.weight}kg` : ""}

```

```

        </p>

```

```

      <div className="space-y-3">

```

```

        {exercise.setResults.map((result, setIndex) => (

```

```

          <div key={setIndex} className="flex items-center gap-3">

```

```

            <span className="text-sm text-gray-600 w-16">Sæt {setIndex + 1}</span>

```

```

            <input

```

```

              type="number"

```

```

      value={result}
      onChange={(e) => updateSetResult(exerciseIndex, setIndex, e.target.value)}
      placeholder="Reps"
      className="w-24 p-3 border rounded text-lg"
      pattern="[0-9]*"
      inputMode="numeric"
    />
    <span className="text-sm text-gray-600">reps</span>
  </div>
  )})
</div>
</div>
  )})

```

```

<div className="fixed bottom-0 left-0 right-0 p-4 bg-white border-t shadow-lg sm:relative
sm:shadow-none sm:border-t-0">
  <div className="flex gap-2 max-w-md mx-auto">
    <button
      onClick={handleComplete}
      className="flex-1 bg-green-500 text-white py-3 px-4 rounded-lg text-lg font-medium
      hover:bg-green-600 flex items-center justify-center gap-2"
    >
      <Save size={24} />
      Gem træning
    </button>
    <button
      onClick={onCancel}

```

```
      className="flex-1 bg-gray-200 text-gray-700 py-3 px-4 rounded-lg text-lg font-medium
      hover:bg-gray-300"
```

```
>
```

```
  Annuller
```

```
</button>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
)
```

```
}
```

export default ActiveWorkout

De vigtigste ændringer er:

1. Fjernet JSX-kommentarer som forårsagede fejlen
2. Tilføjet inputMode="numeric" for et bedre numerisk tastatur på mobil
3. Forbedret styling for touch-interfaces
4. Fastgjort gem/annuller knapper i bunden på mobile enheder

Prøv at opdatere filen med denne kode og fortæl mig om fejlen forsvinder.

CopyRetry

**EW**

Hvor skal den nye fil til eksportfunktionaliteten placeres i hierarkiet?

I dit projekt skal vi oprette en ny mappe kaldet "utils" i src mappen, så strukturen bliver:

Copy

src/

├─ components/ # Dine eksisterende komponenter

├─ utils/ # Ny mappe til hjælpefunktioner

| └─ exportToExcel.js

```
├─ App.jsx
├─ index.css
└─ main.jsx
```

Du kan oprette mappen og filen ved at:

1. Højreklik på src mappen i VS Code
2. Vælg "New Folder"
3. Navngiv den "utils"
4. Højreklik på den nye utils mappe
5. Vælg "New File"
6. Navngiv den "exportToExcel.js"

Skal jeg vise dig indholdet af exportToExcel.js igen?

CopyRetry

**13.10.54 [vite] (client) hmr update /src/App.jsx 13.10.54 [vite] Internal server error:**  
**C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by**  
**Mangaard\Skrivebord\Kodning\Workout Tracker\workout**  
**pasted**

**EW**

Har gennemført ændringerne, men får den her besked, når jeg vil gemme ændringerne i app.jsx

Det ser ud til at vi forsøger at indsætte JSX direkte i filen uden for en komponent. Lad os rette App.jsx så navigationsdelen er inde i komponenten:

jsx

Copy

```
import { useState, useEffect } from 'react'

import ProgramBuilder from './components/ProgramBuilder'

import ActiveWorkout from './components/ActiveWorkout'

import { exportWorkoutToExcel } from './utils/exportToExcel'
```



```

function App() {
  const [view, setView] = useState('programs') // 'programs', 'create', 'workout', 'history'
  const [programs, setPrograms] = useState([])
  const [activeProgram, setActiveProgram] = useState(null)
  const [workoutHistory, setWorkoutHistory] = useState([])

  // Indlæs gemte data
  useEffect(() => {
    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')
    const savedHistory = JSON.parse(localStorage.getItem('workoutHistory') || '[]')
    setPrograms(savedPrograms)
    setWorkoutHistory(savedHistory)
  }, [])

  const handleCompleteWorkout = (workout) => {
    setWorkoutHistory([...workoutHistory, workout])
    localStorage.setItem('workoutHistory', JSON.stringify([...workoutHistory, workout]))
    setActiveProgram(null)
    setView('programs')
  }

  return (
    <div className="min-h-screen bg-gray-50 p-4">
      <h1 className="text-2xl font-bold text-center mb-6">Trænings Tracker</h1>

      {view !== 'workout' && (
        <div className="max-w-md mx-auto mb-6">

```

```

<div className="flex gap-2 flex-wrap">
  <button
    onClick={() => setView('programs')}
    className={`flex-1 py-2 px-4 rounded ${
      view === 'programs'
        ? 'bg-blue-500 text-white'
        : 'bg-gray-200 text-gray-700'
      }`}
  >
    Mine programmer
  </button>
  <button
    onClick={() => setView('create')}
    className={`flex-1 py-2 px-4 rounded ${
      view === 'create'
        ? 'bg-blue-500 text-white'
        : 'bg-gray-200 text-gray-700'
      }`}
  >
    Opret nyt
  </button>
  <button
    onClick={() => setView('history')}
    className={`flex-1 py-2 px-4 rounded ${
      view === 'history'
        ? 'bg-blue-500 text-white'
        : 'bg-gray-200 text-gray-700'
      }`}
  >

```

```

    `}`
  >
  Historik
</button>
</div>
</div>
})

```

```

{view === 'create' && <ProgramBuilder />}

```

```

{view === 'workout' && activeProgram && (
  <ActiveWorkout
    program={activeProgram}
    onComplete={handleCompleteWorkout}
    onCancel={() => {
      setActiveProgram(null)
      setView('programs')
    }}
  />
)}

```

```

{view === 'programs' && (
  <div className="max-w-md mx-auto">
    <div className="space-y-4">
      {programs.map(program => (
        <div key={program.id} className="bg-white p-4 rounded-lg shadow">
          <h3 className="text-lg font-semibold">{program.name}</h3>

```

```

    <p className="text-sm text-gray-600 mb-2">
      {program.exercises.length} øvelser
    </p>

    <button
      onClick={() => {
        setActiveProgram(program)
        setView('workout')
      }}
      className="bg-blue-500 text-white py-1 px-3 rounded text-sm hover:bg-blue-600"
    >
      Start træning
    </button>
  </div>
  )})
</div>
</div>
})

```

```

{view === 'history' && (
  <div className="max-w-md mx-auto">
    <div className="space-y-4">
      {workoutHistory.map((workout, index) => (
        <div key={index} className="bg-white p-4 rounded-lg shadow">
          <div className="flex justify-between items-start mb-2">
            <div>
              <h3 className="text-lg font-semibold">{workout.programName}</h3>
              <p className="text-sm text-gray-600">

```

```

        {new Date(workout.date).toLocaleDateString()}
      </p>
    </div>

    <button
      onClick={() => exportWorkoutToExcel(workout)}
      className="text-blue-500 hover:text-blue-700"
    >
      Eksportér
    </button>
  </div>

  <div className="space-y-2">
    {workout.exercises.map((exercise, exIndex) => (
      <div key={exIndex} className="text-sm">
        <p className="font-medium">{exercise.name}</p>
        <p className="text-gray-600">
          Resultater: {exercise.setResults.join(', ')} reps
        </p>
      </div>
    ))}
  </div>

  </div>

  </div>

  </div>
)

```

```
}
```

```
export default App
```

Dette er den komplette, korrekte version af App.jsx. Den inkluderer:

1. Navigation mellem forskellige views
2. Visning af programmer
3. Oprettelse af nye programmer
4. Aktiv træning
5. Træningshistorik med eksportmulighed

Prøv at erstatte din nuværende App.jsx med denne version og fortæl mig om fejlen forsvinder.

CopyRetry

## EW

Det ser ud til at virke. Jeg får dog lidt underlig formatering (pga dansk alfabet) i excelfilen (se billede)

Ah ja, det er et tegnsætningsproblem. Lad os opdatere exportToExcel.js for at håndtere danske tegn korrekt:

javascript

Copy

```
export const exportWorkoutToExcel = (workout) => {  
  
  // Tilføj BOM for at sikre at Excel læser UTF-8 korrekt  
  
  const BOM = '\uFEFF';
```

```
let csvContent = BOM;
```

```
// Format data for Excel med korrekt dansk encoding
```

```
csvContent += `Træningsprogram: ${workout.programName}\n`;
```

```
csvContent += `Dato: ${new Date(workout.date).toLocaleDateString('da-DK')}\n\n`;
```

```
// Data for hver øvelse
```

```
workout.exercises.forEach(exercise => {
```

```
  csvContent += `${exercise.name}\n`;
```

```
  csvContent += `Mål: ${exercise.sets} sæt × ${exercise.reps} reps`;
```

```
  if (exercise.weight) csvContent += ` @ ${exercise.weight}kg`;
```

```
  csvContent += '\n';
```

```
// Resultater for hvert sæt
```

```
csvContent += "Sæt,Reps udført\n";
```

```
exercise.setResults.forEach((reps, index) => {
```

```
  csvContent += `${index + 1},${reps}\n`;
```

```
});
```

```
csvContent += '\n';
```

```
});
```

```
// Download filen
```

```
const blob = new Blob([csvContent], { type: 'text/csv;charset=utf-8' });
```

```
const url = URL.createObjectURL(blob);
```

```
const link = document.createElement("a");
```

```
link.setAttribute("href", url);
```

```
const fileName = `træning_${workout.programName}_${new
Date(workout.date).toLocaleDateString('da-DK').replace(/\./g, '-')}.csv`;

link.setAttribute("download", fileName);

document.body.appendChild(link);

link.click();

document.body.removeChild(link);

URL.revokeObjectURL(url);

};
```

De vigtigste ændringer er:

1. Tilføjelse af BOM (Byte Order Mark) i starten af filen
2. Brug af `toLocaleDateString('da-DK')` for dansk datoformat
3. Korrekt charset specifikation i Blob oprettelsen
4. Sikrer at filnavnet bruger bindestreger i stedet for punktummer i datoen

Dette burde give bedre håndtering af danske tegn i Excel. Prøv at eksportere igen og fortæl mig om formateringen ser bedre ud.

CopyRetry



## EW

Nu vil jeg gerne arbejde lidt med lidt basal styling af applikationen. Den må gerne ligne denne lidt. Kan du komme med relevante forslag?

Ja, lad os opdatere stylingen til at ligne Nike's trænings-app med et mere moderne og rent look. Her er forslag til ændringer:

1. Først opdater vi det overordnede farvetema i App.jsx:

jsx

Copy

```
<div className="min-h-screen bg-black text-white">

  <div className="max-w-md mx-auto px-4 py-6">

    <h1 className="text-3xl font-bold mb-8">Trænings Tracker</h1>


    { /* Navigation */ }

    <div className="flex gap-1 mb-8 bg-zinc-900 p-1 rounded-xl">

      <button

        onClick={() => setView('programs')}

        className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${

          view === 'programs'

            ? 'bg-white text-black'

            : 'text-gray-400 hover:text-white'

          } ` }

      </button>

    </div>

  </div>
```

```

    >

    Programmer

  </button>

  {/... andre navigation knapper */}

</div>


{/** Program cards */}

<div className="space-y-4">

  {programs.map(program => (

    <div key={program.id} className="bg-zinc-900 rounded-2xl p-6">

      <h3 className="text-xl font-semibold mb-1">{program.name}</h3>

      <p className="text-gray-400 text-sm mb-4">

        {program.exercises.length} øvelser

      </p>

      <button

        onClick={() => startWorkout(program)}

        className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-100
transition-colors"

        >

          Start træning

        </button>

      </div>

    )}}

  </div>

</div>

</div>

```

2. Opdater ActiveWorkout.jsx med et mere moderne look:

jsx

Copy

```
<div className="min-h-screen bg-black text-white p-4">

  <div className="max-w-md mx-auto">

    <h2 className="text-2xl font-bold mb-6">{program.name}</h2>

    {exerciseResults.map((exercise, exerciseIndex) => (

      <div key={exercise.id} className="mb-8 bg-zinc-900 rounded-2xl p-6">

        <h3 className="text-lg font-semibold mb-2">{exercise.name}</h3>

        <p className="text-gray-400 text-sm mb-4">

          Mål: {exercise.sets} sæt × {exercise.reps} reps

          {exercise.weight} && ` @ ${exercise.weight}kg`

        </p>

        <div className="space-y-3">

          {exercise.setResults.map((result, setIndex) => (

            <div key={setIndex} className="flex items-center gap-4">

              <span className="text-gray-400 w-12">Sæt {setIndex + 1}</span>

              <input

                type="number"

                value={result}

                onChange={(e) => updateSetResult(exerciseIndex, setIndex, e.target.value)}

                placeholder="Reps"

                className="w-20 bg-zinc-800 text-white border-0 rounded-lg p-3 text-center"

                pattern="[0-9]*"

                inputMode="numeric"

              />

            </div>

          )

        )

      )

    )

  )

}
```

```

        <span className="text-gray-400">reps</span>
      </div>
    )}]
  </div>
</div>
)]}

```

```

<div className="fixed bottom-0 left-0 right-0 p-4 bg-black border-t border-zinc-900">
  <div className="max-w-md mx-auto flex gap-4">
    <button
      onClick={handleComplete}
      className="flex-1 bg-white text-black py-4 rounded-xl font-medium"
    >
      Afslut træning
    </button>
  </div>
</div>
</div>
</div>

```

### 3. Opdater index.css med nogle globale styles:

css

Copy

@tailwind base;

@tailwind components;

@tailwind utilities;

body {

```
@apply bg-black text-white;

-webkit-font-smoothing: antialiased;

}
```

```
input[type="number"]::-webkit-inner-spin-button,
input[type="number"]::-webkit-outer-spin-button {

  -webkit-appearance: none;

  margin: 0;

}
```

Nøgleændringer inkluderer:

1. Sort baggrund med hvid tekst
2. Afrundede hjørner på alle elementer
3. Brug af zinc-farver til sekundære elementer
4. Større padding og spacing
5. Mere moderne typografi
6. Bedre kontrast og hierarki
7. Renere inputfelter
8. Fast placeret "Afslut træning" knap i bunden

Vil du have mig til at implementere disse ændringer én ad gangen, så vi kan se effekten?

CopyRetry

**EW**

Jeg vil gerne bede om den fulde kode, der skal erstatte den eksisterende i filerne

Her er den fulde kode for hver fil:

1. src/App.jsx:

jsx

Copy

```
import { useState, useEffect } from 'react'
```

```
import ProgramBuilder from './components/ProgramBuilder'
import ActiveWorkout from './components/ActiveWorkout'
import { exportWorkoutToExcel } from './utils/exportToExcel'

function App() {
  const [view, setView] = useState('programs') // 'programs', 'create', 'workout', 'history'
  const [programs, setPrograms] = useState([])
  const [activeProgram, setActiveProgram] = useState(null)
  const [workoutHistory, setWorkoutHistory] = useState([])

  useEffect(() => {
    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')
    const savedHistory = JSON.parse(localStorage.getItem('workoutHistory') || '[]')
    setPrograms(savedPrograms)
    setWorkoutHistory(savedHistory)
  }, [])

  const handleCompleteWorkout = (workout) => {
    const newHistory = [...workoutHistory, workout]
    setWorkoutHistory(newHistory)
    localStorage.setItem('workoutHistory', JSON.stringify(newHistory))
    setActiveProgram(null)
    setView('programs')
  }

  return (
    <div className="min-h-screen bg-black text-white">
```

```
<div className="max-w-md mx-auto px-4 py-6">
```

```
<h1 className="text-3xl font-bold mb-8 text-center">Trænings Tracker</h1>
```

```
{view !== 'workout' && (
```

```
<div className="flex gap-1 mb-8 bg-zinc-900 p-1 rounded-xl">
```

```
<button
```

```
onClick={() => setView('programs')}
```

```
className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
```

```
view === 'programs'
```

```
? 'bg-white text-black'
```

```
: 'text-gray-400 hover:text-white'
```

```
}``
```

```
>
```

```
Programmer
```

```
</button>
```

```
<button
```

```
onClick={() => setView('create')}
```

```
className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
```

```
view === 'create'
```

```
? 'bg-white text-black'
```

```
: 'text-gray-400 hover:text-white'
```

```
}``
```

```
>
```

```
Opret nyt
```

```
</button>
```

```
<button
```

```
onClick={() => setView('history')}
```

```

        className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
          view === 'history'
            ? 'bg-white text-black'
            : 'text-gray-400 hover:text-white'
        }`}
      >
        Historik
      </button>
</div>
)}

```

```

{view === 'create' && <ProgramBuilder />}

```

```

{view === 'workout' && activeProgram && (
  <ActiveWorkout
    program={activeProgram}
    onComplete={handleCompleteWorkout}
    onCancel={() => {
      setActiveProgram(null)
      setView('programs')
    }}
  />
)}

```

```

{view === 'programs' && (
  <div className="space-y-4">
    {programs.map(program => (

```



```

<div key={program.id} className="bg-zinc-900 rounded-2xl p-6">
  <h3 className="text-xl font-semibold mb-1">{program.name}</h3>
  <p className="text-gray-400 text-sm mb-4">
    {program.exercises.length} øvelser
  </p>
  <button
    onClick={() => {
      setActiveProgram(program)
      setView('workout')
    }}
    className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-
100 transition-colors"
  >
    Start træning
  </button>
</div>
)}}
</div>
)}

```

```

{view === 'history' && (
  <div className="space-y-4">
    {workoutHistory.map((workout, index) => (
      <div key={index} className="bg-zinc-900 rounded-2xl p-6">
        <div className="flex justify-between items-start mb-4">
          <div>
            <h3 className="text-xl font-semibold mb-1">{workout.programName}</h3>

```

```
<p className="text-gray-400 text-sm">
  {new Date(workout.date).toLocaleDateString('da-DK')}
</p>
</div>
<button
  onClick={() => exportWorkoutToExcel(workout)}
  className="text-sm text-gray-400 hover:text-white transition-colors"
>
  Eksportér
</button>
</div>
<div className="space-y-3">
  {workout.exercises.map((exercise, exIndex) => (
    <div key={exIndex} className="border-t border-zinc-800 pt-3">
      <p className="font-medium mb-1">{exercise.name}</p>
      <p className="text-gray-400 text-sm">
        Resultater: {exercise.setResults.join(', ')} reps
      </p>
    </div>
  ))}
</div>
</div>
  )}
</div>
</div>
```

```
)  
}
```

export default App

2. src/components/ActiveWorkout.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
import { Save } from 'lucide-react'
```

```
function ActiveWorkout({ program, onComplete, onCancel }) {
```

```
  const [exerciseResults, setExerciseResults] = useState(  
    program.exercises.map(exercise => ({  
      ...exercise,  
      setResults: Array(Number(exercise.sets)).fill("")  
    })))  
}
```

```
const updateSetResult = (exerciseIndex, setIndex, reps) => {
```

```
  setExerciseResults(prev => {
```

```
    const newResults = [...prev]
```

```
    newResults[exerciseIndex] = {
```

```
      ...newResults[exerciseIndex],
```

```
      setResults: newResults[exerciseIndex].setResults.map((set, idx) =>
```

```
        idx === setIndex ? reps : set
```

```
    )
```

```
  }
```

```
    return newResults
  })
}
```

```
const handleComplete = () => {
  const workoutResult = {
    programId: program.id,
    programName: program.name,
    date: new Date().toISOString(),
    exercises: exerciseResults,
  }
  onComplete(workoutResult)
}
```

```
return (
  <div className="min-h-screen bg-black text-white pb-24">
    <div className="max-w-md mx-auto">
      <h2 className="text-2xl font-bold mb-6">{program.name}</h2>

      {exerciseResults.map((exercise, exerciseIndex) => (
        <div key={exercise.id} className="mb-8 bg-zinc-900 rounded-2xl p-6">
          <h3 className="text-lg font-semibold mb-2">{exercise.name}</h3>
          <p className="text-gray-400 text-sm mb-4">
            Mål: {exercise.sets} sæt × {exercise.reps} reps
            {exercise.weight} && ` @ ${exercise.weight}kg`
          </p>

```

```

<div className="space-y-3">
  {exercise.setResults.map((result, setIndex) => (
    <div key={setIndex} className="flex items-center gap-4">
      <span className="text-gray-400 w-12">Sæt {setIndex + 1}</span>
      <input
        type="number"
        value={result}
        onChange={(e) => updateSetResult(exerciseIndex, setIndex, e.target.value)}
        placeholder="Reps"
        className="w-20 bg-zinc-800 text-white border-0 rounded-lg p-3 text-center"
        pattern="[0-9]*"
        inputMode="numeric"
      />
      <span className="text-gray-400">reps</span>
    </div>
  ))}
</div>
</div>
)))}

<div className="fixed bottom-0 left-0 right-0 p-4 bg-black border-t border-zinc-900">
  <div className="max-w-md mx-auto flex gap-4">
    <button
      onClick={handleComplete}
      className="flex-1 bg-white text-black py-4 rounded-xl font-medium"
    >
    Afslut træning
  
```

```
        </button>

        <button
          onClick={onCancel}
          className="flex-1 bg-zinc-900 text-white py-4 rounded-xl font-medium"
        >
          Annuler
        </button>
      </div>
    </div>
  </div>
</div>
)
}
```

export default ActiveWorkout

3. src/index.css:

css

Copy

@tailwind base;

@tailwind components;

@tailwind utilities;

body {

@apply bg-black text-white;

-webkit-font-smoothing: antialiased;

}

```
input[type="number"]::-webkit-inner-spin-button,  
input[type="number"]::-webkit-outer-spin-button {  
  -webkit-appearance: none;  
  margin: 0;  
}
```

```
input[type="number"] {  
  -moz-appearance: textfield;  
}
```

Vil du have mig til at dele koden for de resterende komponenter også (ProgramBuilder og exportToExcel)?

CopyRetry

2 / 2

## EW

Kan du indarbejde muligheden, så man kan uploade et baggrundsbillede/genrebillede til applikationen? Og angive formatet på billedet, så applikationen fortsat er mobiloptimeret?

Ja, lad os tilføje mulighed for et baggrundsbillede i en mobiloptimeret version. Vi kan gøre dette ved at opdatere App.jsx med et hero-område:

jsx

Copy

*// I App.jsx, opdater den yderste div:*

```
function App() {  
  // ... eksisterende state og effects ...  
  
  return (  
    <div className="min-h-screen bg-black text-white relative">  
      {/* Hero baggrundsbillede */}
```

```

<div className="absolute inset-0 z-0">

  

  {/* Gradient overlay for bedre læsbarhed */}

  <div className="absolute inset-0 bg-gradient-to-b from-transparent via-black/70 to-
black" />

</div>

{/* Indhold med z-index for at være over baggrundsbilledet */}

<div className="relative z-10">

  <div className="max-w-md mx-auto px-4 py-6">

    <h1 className="text-3xl font-bold mb-8 text-center text-white drop-shadow-lg">

      Trænings Tracker

    </h1>

    {/* Resten af dit eksisterende indhold */}

    {view !== 'workout' && (

      <div className="flex gap-1 mb-8 bg-zinc-900/80 backdrop-blur-sm p-1 rounded-xl">

        {/* ... eksisterende navigation ... */}

      </div>

    )}

    {/* Opdater program cards med glassmorphism effekt */}

```



```

{view === 'programs' && (
  <div className="space-y-4">
    {programs.map(program => (
      <div key={program.id} className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-
6">
        {/* ... eksisterende program card indhold ... */}
      </div>
    ))}
  </div>
)}

{/* ... resten af dine views ... */}
</div>
</div>
</div>
)
}

```

For at tillade upload af baggrundsbillede, kan vi tilføje en indstillinger-sektion:

jsx

Copy

*// Ny Settings komponent (opret som src/components/Settings.jsx)*

```
import { useState } from 'react'
```

```
function Settings({ onClose }) {
```

```
  const [backgroundImage, setBackgroundImage] = useState(null)
```

```
  const handleImageUpload = (event) => {
```

```
const file = event.target.files[0]

if (file) {
  // Tjek filstørrelse (max 2MB)
  if (file.size > 2 * 1024 * 1024) {
    alert('Billedet må maksimalt være 2MB')
    return
  }

  // Tjek dimensioner
  const img = new Image()
  img.onload = () => {
    if (img.width < 1080 || img.height < 400) {
      alert('Billedet skal minimum være 1080x400 pixels')
      return
    }

    const reader = new FileReader()
    reader.onloadend = () => {
      localStorage.setItem('backgroundImage', reader.result)
      setBackgroundImage(reader.result)
      window.location.reload() // Genindlæs app for at vise nyt baggrundsbillede
    }
    reader.readAsDataURL(file)
  }
  img.src = URL.createObjectURL(file)
}
}
```

```
return (  
  <div className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">  
    <h2 className="text-xl font-semibold mb-4">Indstillinger</h2>  
  
    <div className="space-y-4">  
      <div>  
        <label className="block text-sm font-medium text-gray-400 mb-2">  
          Baggrunds billede  
        </label>  
        <input  
          type="file"  
          accept="image/*"  
          onChange={handleImageUpload}  
          className="w-full text-sm text-gray-400  
            file:mr-4 file:py-2 file:px-4  
            file:rounded-full file:border-0  
            file:text-sm file:font-semibold  
            file:bg-white file:text-black  
            hover:file:bg-gray-100"  
        />  
        <p className="mt-2 text-xs text-gray-500">  
          Anbefalet størrelse: 1080x400 pixels, max 2MB  
        </p>  
      </div>  
  
      <button
```

```

      onClick={onClose}

      className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-100
transition-colors"

    >

      Luk

    </button>

  </div>

</div>

)
}

```

export default Settings

Og tilføj en indstillinger-knap i App.jsx:

jsx

Copy

*// I App.jsx, tilføj ny state*

```
const [showSettings, setShowSettings] = useState(false)
```

*// Tilføj indstillinger-knap i navigationsbaren*

```

<div className="flex gap-1 mb-8 bg-zinc-900/80 backdrop-blur-sm p-1 rounded-xl">

  {/... eksisterende navigation knapper ... */}

  <button

    onClick={() => setShowSettings(true)}

    className="text-gray-400 hover:text-white p-2 rounded-lg"

  >

    <Cog size={20} />

  </button>

```

</div>

*// Tilføj Settings komponenten når showSettings er true*

```
{showSettings && (  
  <Settings onClose={() => setShowSettings(false)} />  
)}
```

Og endelig, hent baggrundsbilledet fra localStorage når appen starter:

jsx

Copy

*// I App.jsx, tilføj til eksisterende useEffect*

```
useEffect(() => {  
  // ... eksisterende kode ...  
  
  // Hent gemt baggrundsbillede  
  const savedImage = localStorage.getItem('backgroundImage')  
  if (savedImage) {  
    document.querySelector('#background-image').src = savedImage  
  }  
}, [])
```

Dette giver:

1. Et mobiloptimeret hero-område med baggrundsbillede
2. Mulighed for at uploade og gemme et tilpasset baggrundsbillede
3. Automatisk validering af billedstørrelse og dimensioner
4. Glassmorphism effekt på UI-elementer for bedre læsbarhed
5. Gradient overlay for at sikre god kontrast med tekst
6. Responsivt design der ser godt ud på alle skærmstørrelser

Vil du have mig til at implementere dette, eller skal jeg justere noget i designet først?

CopyRetry

**EW**

Hvor skal koden indsættes i filerne? Kan du give mig den fulde kode for filerne?

Jeg vil give dig den komplette kode for hver fil:

1. Først, opret en ny fil src/components/Settings.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
function Settings({ onClose }) {
```

```
  const [backgroundImage, setBackgroundImage] = useState(null)
```

```
  const handleImageUpload = (event) => {
```

```
    const file = event.target.files[0]
```

```
    if (file) {
```

```
      if (file.size > 2 * 1024 * 1024) {
```

```
        alert('Billedet må maksimalt være 2MB')
```

```
        return
```

```
      }
```

```
      const img = new Image()
```

```
      img.onload = () => {
```

```
        if (img.width < 1080 || img.height < 400) {
```

```
          alert('Billedet skal minimum være 1080x400 pixels')
```

```
          return
```

```
        }
```

```

const reader = new FileReader()

reader.onloadend = () => {

  localStorage.setItem('backgroundImage', reader.result)

  setBackgroundImage(reader.result)

  window.location.reload()

}

reader.readAsDataURL(file)

}

img.src = URL.createObjectURL(file)

}

}

return (

<div className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">

  <h2 className="text-xl font-semibold mb-4">Indstillinger</h2>

  <div className="space-y-4">

    <div>

      <label className="block text-sm font-medium text-gray-400 mb-2">

        Baggrunds billede

      </label>

      <input

        type="file"

        accept="image/*"

        onChange={handleImageUpload}

        className="w-full text-sm text-gray-400

          file:mr-4 file:py-2 file:px-4

```

```

    file:rounded-full file:border-0
    file:text-sm file:font-semibold
    file:bg-white file:text-black
    hover:file:bg-gray-100"
  />
  <p className="mt-2 text-xs text-gray-500">
    Anbefalet størrelse: 1080x400 pixels, max 2MB
  </p>
</div>

```

```

<button
  onClick={onClose}
  className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-100
transition-colors"
  >
    Luk
  </button>
</div>
</div>
)
}

```

export default Settings

2. Opdater src/App.jsx med den nye kode:

jsx

Copy

```
import { useState, useEffect } from 'react'
```



```

import { Cog } from 'lucide-react'

import ProgramBuilder from './components/ProgramBuilder'

import ActiveWorkout from './components/ActiveWorkout'

import Settings from './components/Settings'

import { exportWorkoutToExcel } from './utils/exportToExcel'


function App() {
  const [view, setView] = useState('programs')
  const [programs, setPrograms] = useState([])
  const [activeProgram, setActiveProgram] = useState(null)
  const [workoutHistory, setWorkoutHistory] = useState([])
  const [showSettings, setShowSettings] = useState(false)


  useEffect(() => {
    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')
    const savedHistory = JSON.parse(localStorage.getItem('workoutHistory') || '[]')

    setPrograms(savedPrograms)

    setWorkoutHistory(savedHistory)


    const savedImage = localStorage.getItem('backgroundImage')
    if (savedImage && document.querySelector('#background-image')) {
      document.querySelector('#background-image').src = savedImage
    }
  }, [])


  const handleCompleteWorkout = (workout) => {
    const newHistory = [...workoutHistory, workout]

```

```
setWorkoutHistory(newHistory)

localStorage.setItem('workoutHistory', JSON.stringify(newHistory))

setActiveProgram(null)

setView('programs')

}
```

```
return (
```

```
<div className="min-h-screen bg-black text-white relative">
```

```
  {/* Hero baggrundsbillede */}
```

```
<div className="absolute inset-0 z-0">
```

```

```

```
<div className="absolute inset-0 bg-gradient-to-b from-transparent via-black/70 to-
black" />
```

```
</div>
```

```
{/* Hovedindhold */}
```

```
<div className="relative z-10">
```

```
<div className="max-w-md mx-auto px-4 py-6">
```

```
<h1 className="text-3xl font-bold mb-8 text-center text-white drop-shadow-lg">
```

```
  Trænings Tracker
```

```
</h1>
```

```

{view !== 'workout' && (
  <div className="flex items-center gap-1 mb-8 bg-zinc-900/80 backdrop-blur-sm p-1
rounded-xl">
    <button
      onClick={() => setView('programs')}
      className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
        view === 'programs'
          ? 'bg-white text-black'
          : 'text-gray-400 hover:text-white'
        }`}
    >
      Programmer
    </button>
    <button
      onClick={() => setView('create')}
      className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
        view === 'create'
          ? 'bg-white text-black'
          : 'text-gray-400 hover:text-white'
        }`}
    >
      Opret nyt
    </button>
    <button
      onClick={() => setView('history')}
      className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
        view === 'history'

```

```

        ? 'bg-white text-black'
        : 'text-gray-400 hover:text-white'
    }` }
  >
  Historik
</button>

<button
  onClick={() => setShowSettings(true)}
  className="text-gray-400 hover:text-white p-2 rounded-lg"
>

  <Cog size={20} />

</button>

</div>
)}

```

```

{showSettings && (
  <Settings onClose={() => setShowSettings(false)} />
)}

```

```

/* Eksisterende views */
/* ... resten af din eksisterende kode for forskellige views ... */

```

```

</div>

```

```

</div>

```

```

</div>

```

```

)

```

```

}

```

export default App

Sørg for at installere lucide-react hvis du ikke allerede har det:

bash

Copy

npm install lucide-react

De vigtigste ændringer er:

1. Tilføjelse af Settings komponenten
2. Nyt baggrundsbillede layout
3. Glassmorphism effekter på UI elementer
4. Indstillinger-knap i navigation
5. Håndtering af baggrundsbillede i localStorage

Vil du have mig til at dele hele koden for de andre komponenter også, eller skal vi først teste om dette virker?

CopyRetry

**EW**

Hvordan uploader jeg et billede til applikationen, så jeg kan teste det?

For at teste billede-upload funktionen, skal vi:

1. Klikke på tandhjuls-ikonet (Cog) i navigationsbaren, som åbner indstillingerne
2. I indstillingerne finder du en "Choose File" eller "Vælg fil" knap
3. Vælg et billede fra din computer som:
  - Er mindre end 2MB
  - Har en opløsning på mindst 1080x400 pixels

Men jeg kan se vi skal tilføje en midlertidig løsning for placeholder billedet, da /api/placeholder/1080/400 ikke eksisterer i vores setup.

Lad os opdatere img-taggen i App.jsx til at bruge et standard baggrundsbillede:

jsx

Copy

```
<img  
  id="background-image"  
  src={localStorage.getItem('backgroundImage') || 'https://images.unsplash.com/photo-1534438327276-14e5300c3a48?q=80&w=1080&h=400&fit=crop'}  
  alt="Background"  
  className="w-full h-[400px] object-cover brightness-50"  
>
```

Denne ændring giver os et standard træningsbillede fra Unsplash, som vil blive erstattet når brugeren uploader deres eget billede.

Vil du have mig til at dele den opdaterede kode, så du kan teste billede-upload funktionen?

CopyRetry

## EW

Virker de her billeddimensioner rigtige? Det er bredformat. Burde det ikke være højformat?

For en mobiloptimeret applikation ville et højformat-billede faktisk være mere hensigtsmæssigt. Det nuværende bredformatbillede (1080x400) betyder, at meget af billedet bliver beskåret på en mobilskærm, eller at billedet bliver meget lille i højden.

Lad os ændre dimensionerne til et mere mobil-venligt format, f.eks. 1080x1350 (9:16 ratio, som er optimalt for mobilvisning). Dette vil give et bedre visuelt udtryk på mobilenheder.

Lad os opdatere koden i App.jsx:

jsx

Copy

```
<img  
  id="background-image"
```

```
src={localStorage.getItem('backgroundImage') || 'https://images.unsplash.com/photo-1534438327276-14e5300c3a48?q=80&w=1080&h=1350&fit=crop'}
```

```
alt="Background"
```

```
className="w-full h-[100vh] object-cover brightness-50"
```

```
/>
```

Og i Settings.jsx skal vi også opdatere valideringen:

```
jsx
```

Copy

```
img.onload = () => {
```

```
  if (img.width < 1080 || img.height < 1350) {
```

```
    alert('Billedet skal minimum være 1080x1350 pixels')
```

```
    return
```

```
  }
```

```
  // ... resten af koden
```

```
}
```

Denne ændring vil:

1. Give bedre udnyttelse af mobil-skærmens højde
2. Skabe et mere immersivt design på mobilenheder
3. Give mere plads til at vise motiverende træningsbilleder
4. Reducere behovet for at beskære billeder u hensigtsmæssigt

Skal jeg dele den komplette opdaterede kode for begge filer?

CopyRetry

**EW**

Kan du give mig hele koden for begge filer`?

Her er den komplette, opdaterede kode for begge filer:

1. src/App.jsx:

```
jsx
```

Copy

```
import { useState, useEffect } from 'react'

import { Cog } from 'lucide-react'

import ProgramBuilder from './components/ProgramBuilder'

import ActiveWorkout from './components/ActiveWorkout'

import Settings from './components/Settings'

import { exportWorkoutToExcel } from './utils/exportToExcel'

function App() {

  const [view, setView] = useState('programs')

  const [programs, setPrograms] = useState([])

  const [activeProgram, setActiveProgram] = useState(null)

  const [workoutHistory, setWorkoutHistory] = useState([])

  const [showSettings, setShowSettings] = useState(false)

  useEffect(() => {

    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms') || '[]')

    const savedHistory = JSON.parse(localStorage.getItem('workoutHistory') || '[]')

    setPrograms(savedPrograms)

    setWorkoutHistory(savedHistory)

    const savedImage = localStorage.getItem('backgroundImage')

    if (savedImage && document.querySelector('#background-image')) {

      document.querySelector('#background-image').src = savedImage

    }

  }, [])

}
```



```

const handleCompleteWorkout = (workout) => {
  const newHistory = [...workoutHistory, workout]
  setWorkoutHistory(newHistory)
  localStorage.setItem('workoutHistory', JSON.stringify(newHistory))
  setActiveProgram(null)
  setView('programs')
}

```

```

return (
  <div className="min-h-screen bg-black text-white relative">
    /* Hero baggrundsbillede */
    <div className="absolute inset-0 z-0">
      <img
        id="background-image"
        src={localStorage.getItem('backgroundImage') || 'https://images.unsplash.com/photo-1534438327276-14e5300c3a48?q=80&w=1080&h=1350&fit=crop'}
        alt="Background"
        className="w-full h-[100vh] object-cover brightness-50"
      />
      <div className="absolute inset-0 bg-gradient-to-b from-transparent via-black/70 to-black" />
    </div>

    /* Hovedindhold */
    <div className="relative z-10">
      <div className="max-w-md mx-auto px-4 py-6">
        <h1 className="text-3xl font-bold mb-8 text-center text-white drop-shadow-lg">

```

Trænings Tracker

</h1>

{view !== 'workout' && (

<div className="flex items-center gap-1 mb-8 bg-zinc-900/80 backdrop-blur-sm p-1 rounded-xl">

<button

onClick={() => setView('programs')}

className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors \${

view === 'programs'

? 'bg-white text-black'

: 'text-gray-400 hover:text-white'

}`}

>

Programmer

</button>

<button

onClick={() => setView('create')}

className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors \${

view === 'create'

? 'bg-white text-black'

: 'text-gray-400 hover:text-white'

}`}

>

Opret nyt

</button>

<button

```

onClick={() => setView('history')}

className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
  view === 'history'
    ? 'bg-white text-black'
    : 'text-gray-400 hover:text-white'
}`}
>
  Historik
</button>
<button
  onClick={() => setShowSettings(true)}
  className="text-gray-400 hover:text-white p-2 rounded-lg"
>
  <Cog size={20} />
</button>
</div>
)}

{showSettings && (
  <Settings onClose={() => setShowSettings(false)} />
)}

{view === 'create' && <ProgramBuilder />}

{view === 'workout' && activeProgram && (
  <ActiveWorkout
    program={activeProgram}

```

```

    onComplete={handleCompleteWorkout}
    onCancel={() => {
      setActiveProgram(null)
      setView('programs')
    }}
  />
)}

```

```

{view === 'programs' && (
  <div className="space-y-4">
    {programs.map(program => (
      <div key={program.id} className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-
6">
        <h3 className="text-xl font-semibold mb-1">{program.name}</h3>
        <p className="text-gray-400 text-sm mb-4">
          {program.exercises.length} øvelser
        </p>
        <button
          onClick={() => {
            setActiveProgram(program)
            setView('workout')
          }}
          className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-
100 transition-colors"
        >
          Start træning
        </button>
      </div>
    ))}
  </div>
)}

```

```

    </div>

  )}

</div>

)}

{view === 'history' && (
  <div className="space-y-4">
    {workoutHistory.map((workout, index) => (
      <div key={index} className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">
        <div className="flex justify-between items-start mb-4">
          <div>
            <h3 className="text-xl font-semibold mb-1">{workout.programName}</h3>
            <p className="text-gray-400 text-sm">
              {new Date(workout.date).toLocaleDateString('da-DK')}
            </p>
          </div>
          <button
            onClick={() => exportWorkoutToExcel(workout)}
            className="text-sm text-gray-400 hover:text-white transition-colors"
          >
            Eksportér
          </button>
        </div>
        <div className="space-y-3">
          {workout.exercises.map((exercise, exIndex) => (
            <div key={exIndex} className="border-t border-zinc-800 pt-3">
              <p className="font-medium mb-1">{exercise.name}</p>

```

```

    <p className="text-gray-400 text-sm">
      Resultater: {exercise.setResults.join(', ')} reps
    </p>
  </div>

  )}
</div>

</div>

  )}
</div>

  )}
</div>

</div>
)
}

```

export default App

2. src/components/Settings.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
function Settings({ onClose }) {
```

```
  const [backgroundImage, setBackgroundImage] = useState(null)
```

```
  const handleImageUpload = (event) => {
```

```
    const file = event.target.files[0]
```

```

if (file) {
  if (file.size > 2 * 1024 * 1024) {
    alert('Billedet må maksimalt være 2MB')
    return
  }

  const img = new Image()
  img.onload = () => {
    if (img.width < 1080 || img.height < 1350) {
      alert('Billedet skal minimum være 1080x1350 pixels')
      return
    }

    const reader = new FileReader()
    reader.onloadend = () => {
      localStorage.setItem('backgroundImage', reader.result)
      setBackgroundImage(reader.result)
      window.location.reload()
    }
    reader.readAsDataURL(file)
  }
  img.src = URL.createObjectURL(file)
}

return (
  <div className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">

```

```
<h2 className="text-xl font-semibold mb-4">Indstillinger</h2>
```

```
<div className="space-y-4">
```

```
<div>
```

```
<label className="block text-sm font-medium text-gray-400 mb-2">
```

```
  Baggrunds billede
```

```
</label>
```

```
<input
```

```
  type="file"
```

```
  accept="image/*"
```

```
  onChange={handleImageUpload}
```

```
  className="w-full text-sm text-gray-400
```

```
  file:mr-4 file:py-2 file:px-4
```

```
  file:rounded-full file:border-0
```

```
  file:text-sm file:font-semibold
```

```
  file:bg-white file:text-black
```

```
  hover:file:bg-gray-100"
```

```
/>
```

```
<p className="mt-2 text-xs text-gray-500">
```

```
  Anbefalet størrelse: 1080x1350 pixels, max 2MB
```

```
</p>
```

```
</div>
```

```
<button
```

```
  onClick={onClose}
```

```
  className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-100  
transition-colors"
```



```
>  
  Luk  
</button>  
</div>  
</div>  
)  
}
```

export default Settings

Nu burde du kunne:

1. Se baggrundsbilledet i fuld højde på mobil
2. Åbne indstillinger via tandhjuls-ikonet
3. Uploade et nyt baggrundsbillede der passer til mobilvisning
4. Se den korrekte glassmorphism effekt på alle elementer

CopyRetry

**EW**

Er tandhjulsikonet vi VS Code?

Nej, vi skal installere ikonet fra lucide-react pakken. Kør følgende kommando i VS Code terminalen:

bash

Copy

npm install lucide-react

Dette vil installere ikonpakken som indeholder tandhjulsikonet (Cog) og andre ikoner vi kan bruge i applikationen.

Efter installationen burde ikonet automatisk være tilgængeligt, da vi allerede har import-linjen i App.jsx:

jsx

Copy

```
import { Cog } from 'lucide-react'
```

Fortæl mig hvis du får nogen fejl under installationen af pakken.

CopyRetry

## EW

Får den her besked, når jeg vil uploade et foto

Dette er fordi vores valideringskrav for billedet er sat for højt. Lad os justere størrelsen ned til noget mere realistisk. Vi skal opdatere Settings.jsx:

jsx

Copy

*// I Settings.jsx, ændre valideringen til mere realistiske størrelse:*

```
const handleImageUpload = (event) => {  
  const file = event.target.files[0]  
  if (file) {  
    if (file.size > 5 * 1024 * 1024) { // Øget til 5MB  
      alert('Billedet må maksimalt være 5MB')  
      return  
    }  
  
    const img = new Image()  
    img.onload = () => {  
      if (img.width < 640 || img.height < 960) { // Reduceret minimumskrav
```

```

    alert('Billedet skal minimum være 640x960 pixels')
    return
  }

  const reader = new FileReader()
  reader.onloadend = () => {
    localStorage.setItem('backgroundImage', reader.result)
    setBackgroundImage(reader.result)
    window.location.reload()
  }
  reader.readAsDataURL(file)
}
img.src = URL.createObjectURL(file)
}
}

```

Og opdater også informationsteksten i komponenten:

jsx

Copy

```
<p className="mt-2 text-xs text-gray-500">
```

Anbefalet størrelse: minimum 640x960 pixels, max 5MB

```
</p>
```

Dette giver mere realistiske krav til billedstørrelsen som de fleste mobilfotos vil kunne opfylde. Prøv at uploade dit billede igen efter denne ændring.

CopyRetry

**EW**

Kan du give mig den fulde kode til filerne?

Her er den komplette, opdaterede kode for begge filer:

1. src/App.jsx:

jsx

Copy

```
import { useState, useEffect } from 'react'
import { Cog } from 'lucide-react'
import ProgramBuilder from './components/ProgramBuilder'
import ActiveWorkout from './components/ActiveWorkout'
import Settings from './components/Settings'
import { exportWorkoutToExcel } from './utils/exportToExcel'

function App() {
  const [view, setView] = useState('programs')
  const [programs, setPrograms] = useState([])
  const [activeProgram, setActiveProgram] = useState(null)
  const [workoutHistory, setWorkoutHistory] = useState([])
  const [showSettings, setShowSettings] = useState(false)

  useEffect(() => {
    const savedPrograms = JSON.parse(localStorage.getItem('workoutPrograms')) || '[]'
    const savedHistory = JSON.parse(localStorage.getItem('workoutHistory')) || '[]'
    setPrograms(savedPrograms)
    setWorkoutHistory(savedHistory)

    const savedImage = localStorage.getItem('backgroundImage')
    if (savedImage && document.querySelector('#background-image')) {
      document.querySelector('#background-image').src = savedImage
    }
  })
}
```

```
}, [])
```

```
const handleCompleteWorkout = (workout) => {  
  const newHistory = [...workoutHistory, workout]  
  setWorkoutHistory(newHistory)  
  localStorage.setItem('workoutHistory', JSON.stringify(newHistory))  
  setActiveProgram(null)  
  setView('programs')  
}
```

```
return (  
  <div className="min-h-screen bg-black text-white relative">  
    {/* Hero baggrundsbillede */}  
    <div className="absolute inset-0 z-0">  
      <img  
        id="background-image"  
        src={localStorage.getItem('backgroundImage') || 'https://images.unsplash.com/photo-1534438327276-14e5300c3a48?q=80&w=1080&h=1350&fit=crop'}  
        alt="Background"  
        className="w-full h-[100vh] object-cover brightness-50"  
      />  
      <div className="absolute inset-0 bg-gradient-to-b from-transparent via-black/70 to-black" />  
    </div>  
  
    {/* Hovedindhold */}  
    <div className="relative z-10">
```

```
<div className="max-w-md mx-auto px-4 py-6">
```

```
<h1 className="text-3xl font-bold mb-8 text-center text-white drop-shadow-lg">
```

```
  Trænings Tracker
```

```
</h1>
```

```
{view !== 'workout' && (
```

```
  <div className="flex items-center gap-1 mb-8 bg-zinc-900/80 backdrop-blur-sm p-1 rounded-xl">
```

```
    <button
```

```
      onClick={() => setView('programs')}
```

```
      className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
```

```
        view === 'programs'
```

```
        ? 'bg-white text-black'
```

```
        : 'text-gray-400 hover:text-white'
```

```
      }`}
```

```
    >
```

```
      Programmer
```

```
  </button>
```

```
  <button
```

```
    onClick={() => setView('create')}
```

```
    className={`flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
```

```
      view === 'create'
```

```
      ? 'bg-white text-black'
```

```
      : 'text-gray-400 hover:text-white'
```

```
    }`}
```

```
  >
```

```
    Opret nyt
```

```

</button>

<button
  onClick={() => setView('history')}
  className={` flex-1 py-3 px-4 rounded-lg text-sm font-medium transition-colors ${
    view === 'history'
      ? 'bg-white text-black'
      : 'text-gray-400 hover:text-white'
    }` }
>
  Historik
</button>

<button
  onClick={() => setShowSettings(true)}
  className="text-gray-400 hover:text-white p-2 rounded-lg"
>
  <Cog size={20} />
</button>
</div>
)}

{showSettings && (
  <Settings onClose={() => setShowSettings(false)} />
)}

{view === 'create' && <ProgramBuilder />}

{view === 'workout' && activeProgram && (

```

```

<ActiveWorkout
  program={activeProgram}
  onComplete={handleCompleteWorkout}
  onCancel={() => {
    setActiveProgram(null)
    setView('programs')
  }}
/>
)}

```

```

{view === 'programs' && (
  <div className="space-y-4">
    {programs.map(program => (
      <div key={program.id} className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-
6">
        <h3 className="text-xl font-semibold mb-1">{program.name}</h3>
        <p className="text-gray-400 text-sm mb-4">
          {program.exercises.length} øvelser
        </p>
        <button
          onClick={() => {
            setActiveProgram(program)
            setView('workout')
          }}
          className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-
100 transition-colors"
        >

```



```

        Start træning
      </button>
    </div>
  )}
</div>
)}

{view === 'history' && (
  <div className="space-y-4">
    {workoutHistory.map((workout, index) => (
      <div key={index} className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">
        <div className="flex justify-between items-start mb-4">
          <div>
            <h3 className="text-xl font-semibold mb-1">{workout.programName}</h3>
            <p className="text-gray-400 text-sm">
              {new Date(workout.date).toLocaleDateString('da-DK')}
            </p>
          </div>
          <button
            onClick={() => exportWorkoutToExcel(workout)}
            className="text-sm text-gray-400 hover:text-white transition-colors"
          >
            Eksportér
          </button>
        </div>
        <div className="space-y-3">
          {workout.exercises.map((exercise, exIndex) => (

```

```

    <div key={exIndex} className="border-t border-zinc-800 pt-3">
      <p className="font-medium mb-1">{exercise.name}</p>
      <p className="text-gray-400 text-sm">
        Resultater: {exercise.setResults.join(', ')} reps
      </p>
    </div>

  )}
</div>
</div>

  )}
</div>

  )}
</div>
</div>
)
}

```

export default App

2. src/components/Settings.jsx:

jsx

Copy

```
import { useState } from 'react'
```

```
function Settings({ onClose }) {
```

```
  const [backgroundImage, setBackgroundImage] = useState(null)
```

```
const handleImageUpload = (event) => {  
  const file = event.target.files[0]  
  if (file) {  
    if (file.size > 5 * 1024 * 1024) { // 5MB  
      alert('Billedet må maksimalt være 5MB')  
      return  
    }  
  
    const img = new Image()  
    img.onload = () => {  
      if (img.width < 640 || img.height < 960) {  
        alert('Billedet skal minimum være 640x960 pixels')  
        return  
      }  
  
      const reader = new FileReader()  
      reader.onloadend = () => {  
        localStorage.setItem('backgroundImage', reader.result)  
        setBackgroundImage(reader.result)  
        window.location.reload()  
      }  
      reader.readAsDataURL(file)  
    }  
    img.src = URL.createObjectURL(file)  
  }  
}
```

```
return (  
  
  <div className="bg-zinc-900/80 backdrop-blur-sm rounded-2xl p-6">  
  
    <h2 className="text-xl font-semibold mb-4">Indstillinger</h2>  
  
    <div className="space-y-4">  
  
      <div>  
  
        <label className="block text-sm font-medium text-gray-400 mb-2">  
          Baggrunds billede  
        </label>  
  
        <input  
          type="file"  
          accept="image/*"  
          onChange={handleImageUpload}  
          className="w-full text-sm text-gray-400  
            file:mr-4 file:py-2 file:px-4  
            file:rounded-full file:border-0  
            file:text-sm file:font-semibold  
            file:bg-white file:text-black  
            hover:file:bg-gray-100"  
        />  
  
        <p className="mt-2 text-xs text-gray-500">  
          Anbefalet størrelse: minimum 640x960 pixels, max 5MB  
        </p>  
      </div>  
  
      <button  
        onClick={onClose}
```

```
        className="w-full bg-white text-black py-3 rounded-xl font-medium hover:bg-gray-100 transition-colors"
```

```
    >
```

```
        Luk
```

```
    </button>
```

```
</div>
```

```
</div>
```

```
)
```

```
}
```

export default Settings

De vigtigste ændringer er:

1. Reduceret minimumskrav til billedstørrelse til 640x960 pixels
2. Øget maksimal filstørrelse til 5MB
3. Opdateret hjælpeteksten til at afspejle de nye krav

Prøv nu at uploade dit billede igen - det burde virke med de nye, mere realistiske krav.

CopyRetry

## EW

Pt. kører webapplikationen lokalt på min computer. Hvad skal jeg gøre, hvis jeg gerne vil teste den på min mobiltelefon?

For at teste webapplikationen på din mobiltelefon har du flere muligheder:

### 1. **Nem løsning - Samme WiFi netværk:**

- Find din computers lokale IP-adresse ved at åbne terminal/cmd og skrive:

```
bash
```

Copy

```
ipconfig    # Windows
```

```
ifconfig    # Mac/Linux
```

- Find IP-adressen (ser typisk ud som 192.168.x.x)
- På din mobiltelefon, åbn browseren og indtast:

Copy

http://[din.ip.adresse]:5173

- Din computer og mobiltelefon skal være på samme WiFi-netværk

## 2. Alternativt kan du bruge Vite's --host flag:

- Stop din nuværende server (Ctrl+C i terminalen)
- Åbn package.json og ændr "dev" scriptet til:

json

Copy

```
"scripts": {  
  "dev": "vite --host"  
}
```

- Start serveren igen med npm run dev
- Vite vil nu vise en netværks-URL du kan bruge på din mobil

## 3. Professionel løsning - Deploy til hosting service:

- Du kan deploye til gratis tjenester som:
  - Netlify
  - Vercel
  - GitHub Pages

Vil du have mig til at guide dig gennem en af disse metoder?

CopyRetry

**EW**

Er det i kommandoprompten på min computer, at jeg skal finde ip-adressen?

Ja, du kan følge disse trin:

1. Åbn kommandoprompt (cmd) på din computer:
  - Tryk Windows + R

- Skriv "cmd" og tryk Enter

2. I kommandoprompten, skriv:

Copy

ipconfig

3. Find linjen der hedder "IPv4-adresse" eller "IPv4 Address" under dit WiFi-netværk. Det er typisk et nummer der ser sådan ud: 192.168.1.x
4. Når du har fundet IP-adressen, kan du på din mobil (som skal være på samme WiFi) åbne en browser og skrive:

Copy

http://192.168.1.x:5173

(Erstat 192.168.1.x med din faktiske IP-adresse)

Vil du have mig til at guide dig gennem processen?

CopyRetry

**EW**

Indtastede i VS Code og fik denne melding PS C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by Mangaard\Skrivebord\Kodning\Workout Tracker\workout-tracker> ipconfig # Windows >> ifconfig # Mac/Linux Windows IP Configuration Wireless LAN adapter LAN-forbindelse\* 1: Media State . . . . . : Media disconnected Connection-specific DNS Suffix . : Wireless LAN adapter LAN-forbindelse\* 2: Media State . . . . . : Media disconnected Connection-specific DNS Suffix . : Wireless LAN adapter Wi-Fi: Connection-specific DNS Suffix . : Link-local IPv6 Address . . . . : fe80::3f9b:bdf1:7d76:5bd8%16 IPv4 Address. . . . . : 192.168.68.54 Subnet Mask . . . . . : 255.255.252.0 Default Gateway . . . . . : 192.168.68.1 Ethernet adapter Bluetooth-netværksforbindelse: Media State . . . . . : Media disconnected Connection-specific DNS Suffix . : Ethernet adapter vEthernet (Default Switch): Connection-specific DNS Suffix . : Link-local IPv6 Address . . . . : fe80::1799:af53:deb1:e9c4%32 IPv4 Address. . . . . : 172.19.112.1 Subnet Mask . . . . . : 255.255.240.0 Default Gateway . . . . . : ifconfig: The term 'ifconfig' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again. At line:2 char:1 + ifconfig # Mac/Linux + ~~~~~ + CategoryInfo : ObjectNotFound: (ifconfig:String) [], CommandNotFoundException + FullyQualifiedErrorId : CommandNotFoundException PS C:\Users\EsbenWürtzSørensen\OneDrive - PUBLICO - powered by Mangaard\Skrivebord\Kodning\Workout Tracker\workout-tracker>

Din IP-adresse er 192.168.68.54 - du kan se den under "Wi-Fi" adapter, hvor der står "IPv4 Address".

Så nu kan du på din mobil (mens den er på samme WiFi-netværk som din computer):

1. Åbne din browser
2. Indtaste denne adresse:

Copy

`http://192.168.68.54:5173`

Men først skal vi sikre at Vite serveren kører med host flag. I VS Code:

1. Stop din nuværende server hvis den kører (Ctrl+C i terminalen)
2. Åbn package.json og ændr "dev" linjen til:

json

Copy

```
"scripts": {  
  "dev": "vite --host"  
}
```

3. Start serveren igen med:

bash

Copy

`npm run dev`

Nu burde du kunne tilgå siden fra din mobil med den IP-adresse jeg nævnte ovenfor. Fortæl mig hvis du har brug for hjælp med nogle af trinnene.

CopyRetry

**EW**

kan du give mig hele koden til package.json?

Ja, her er den komplette package.json fil:

json

Copy



```
{
  "name": "workout-tracker",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "scripts": {
    "dev": "vite --host",
    "build": "vite build",
    "lint": "eslint . --ext js,jsx --report-unused-disable-directives --max-warnings 0",
    "preview": "vite preview"
  },
  "dependencies": {
    "lucide-react": "^0.312.0",
    "react": "^18.2.0",
    "react-dom": "^18.2.0"
  },
  "devDependencies": {
    "@types/react": "^18.2.43",
    "@types/react-dom": "^18.2.17",
    "@vitejs/plugin-react": "^4.2.1",
    "autoprefixer": "^10.4.16",
    "eslint": "^8.55.0",
    "eslint-plugin-react": "^7.33.2",
    "eslint-plugin-react-hooks": "^4.6.0",
    "eslint-plugin-react-refresh": "^0.4.5",
    "postcss": "^8.4.32",
    "tailwindcss": "^3.4.0",
  }
}
```

```
"vite": "^5.0.8"
```

```
}
```

```
}
```

Efter du har opdateret filen, skal du:

1. Gemme filen
2. Stoppe serveren hvis den kører (Ctrl+C)
3. Starte serveren igen med npm run dev

Derefter burde du kunne tilgå siden fra din mobil ved at gå til <http://192.168.68.54:5173> i mobilens browser.

CopyRetry

[Claude can make mistakes. Please double-check responses.](#)

3.5 Sonnet

Choose style

2

Chat controls

3.5 Sonnet

Most intelligent model [Learn more](#)

Artifacts

- Trænings Tracker App

Click to open component • 4 versions

- System Components Specification

Click to open code • 1 version

Content

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

- 

image.png

- 

image.png

- 

image.png

- 

image.png

-

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

- 

image.png

- 

image.png

- 

image.png



•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

•

image.png

- Pasted content

10.98 KB •59 lines

- Pasted content

11.11 KB •60 lines

- Pasted content

11.80 KB •69 lines

- Pasted content

5.98 KB •49 lines

Chat styles

Font

DefaultMatch systemDyslexic friendly

Customizable Workout Tracking App - Claude