

# Relazione React

## *Sleep Up* (gioco di parole con Sleep App)

Sleep Up è un'applicazione web sviluppata con React che permette di visualizzare e analizzare dati sul sonno prelevati da un file CSV generato da un dispositivo wearable. L'interfaccia principale consente di selezionare una data, e mostrare statistiche, grafici e suggerimenti relativi al sonno di quella notte.

L'app è strutturata in modo modulare, con più componenti React dedicati al caricamento dati, calcolo dei risultati e rappresentazioni grafiche.

---

### Struttura dell'app:

L'applicazione ha una struttura tipica delle applicazioni create con **create-react-app**, ovvero un comando da prompt dei comandi, che permette a React di generare un esempio funzionante.

Sleep\_Up-main/

- ├─ README.md
- ├─ package.json
- ├─ package-lock.json
- ├─ RelazioneUniVR.pdf
- ├─ public/
  - ├─ index.html
  - └─ data/
    - └─ 4-sleep\_data\_2025-02-11.csv
- └─ src/
  - ├─ App.js
  - ├─ index.css
  - ├─ index.js
  - ├─ components/
    - ├─ DataLoader.jsx
    - ├─ SleepChartPie.jsx
    - ├─ SleepChartTimeline.jsx
    - ├─ SleepScoreCard.jsx
    - ├─ SleepTrendsCombined.jsx
    - └─ Tips.jsx
  - └─ utils/
    - ├─ CalculateScore.js
    - ├─ FormatDuration.js
    - └─ ParseCsv.js

## Flusso dei dati:

1. L'utente seleziona una data (da `<input type="date">`).
  2. Il componente `App.js` aggiorna il nome del file CSV  
(es. `4-sleep_data_2025-02-11.csv`) e lo passa al `DataLoader`.
  3. Il `DataLoader` usa *Papaparse* per leggere il file e utilizza la funzione callback `onData(data)` di `App` per passargli i dati.
  4. L'app chiama un componente `Calculate Score` che contiene la funzione `analyzeSleep`, che calcola punteggi e statistiche.
  5. Il componente `App` distribuisce i risultati ai componenti figli (`SleepScoreCard`, per creare un riepilogo del punteggio e dell'analisi, `SleepChartPie` e `SleepChartTimeline`, per creare dei grafici.).
- 

## Routing:

L'app è una **single-page application** senza routing.

Tutti i contenuti si aggiornano dinamicamente all'interno di un'unica vista.

---

# Componenti React principali

## App.js

- Gestisce lo stato globale: `selectedDate`, `analysis`, `allRecords`.
- Include l'input data, `DataLoader` e tutti i componenti di visualizzazione.
- Smista i dati ai componenti figli tramite props.

### Esempio:

```
<DataLoader filename={`4-sleep_data_${selectedDate}.csv`}
onData={handleData} />

{analysis ? (
  <div className="analysis-card">
    <SleepScoreCard analysis={analysis} />
    <SleepChartPie data={analysis} />
    <SleepChartTimeline data={analysis.rawData} />
    <Tips analysis={analysis} />
  </div>
) : (
  <p style={{ marginTop: '1rem' }}>Nessun dato disponibile per
questa data.</p>
)
```

---

## DataLoader.jsx

- Usa `Papaparse` per caricare e interpretare i file CSV.
- Passa i dati analizzati a `App` tramite `onData`.

### Esempio:

```
export default function DataLoader({ filename, onData })
```

---

## ParseCsv.js

- Genera una funzione asincrona che risolva o rigetti la prop in base all'esito del caricamento del file CSV
- Utilizza *Papaparse* per restituire un array di oggetti tramite una map:

Esempio:

```
.map(r => ( //conversione in date e valori
  {
    timestamp: new Date(r['Timestamp']),
    stage: r['Sleep Stage']
  }));
```

---

## CalculateScore.js

In CalculateScore c'è la funzione *analyzeSleep(records)*:

- Conta i minuti per ogni fase del sonno.
- Calcola punteggio (score) in base alle regole predefinite.
- Restituisce un oggetto con:
  - *minuteCounts* (conto dei minuti)
  - *totalMin* (totale dei minuti)
  - *score* (punteggio calcolato)
  - *interpretation* (interpretazione)
  - *rawData* (dati grezzi per i grafici)

Esempio:

```
Count : Light: 120, Deep: 90, REM: 60, Awake: 30, unknown: 0
```

---

## Componenti di visualizzazione

### SleepScoreCard.jsx

Restituisce i risultati in maniera testuale

Esempio:

```
return (  
  <div className="SleepCard">  
    <h2>Risultati Analisi</h2>  
  
    <p><strong>Ore totali dormite:</strong>  
{formatDuration(analysis.totalMin)}</p> { /* aggiustato */}  
  
    <p><strong>Punteggio:</strong> {analysis.score}/100</p>  
  );
```

---

### SleepChartPie.jsx & SleepChartTimeline.jsx

Mostrano un grafico a torta e un grafico a linee per rispettivamente: la distribuzione delle fasi e l'andamento temporale delle fasi del sonno, usando la libreria [Recharts](#).

SleepChartTimeline converte anche le fasi in valori numerici

```
const sleepStageMap = {Awake: 0, Light: 1, REM: 2, Deep: 3,};  
// mappa le fasi del sonno in valori numerici per rappresentarle  
sull'asse Y
```

e sull'asse delle x inserisce il tempo in cui avviene il sonno.

---

## SleepTrendsCombined.jsx

- Mostra l'**andamento del sonno** (punteggio o durata) per settimana o mese.
- Consente di scegliere tra:
  - Vista settimanale / mensile
  - Punteggio / durata
- Usa un **BarChart** di Recharts.

### Esempio:

```
<BarChart margin={{ top: 10, right: 10, left: 35, bottom: 10 }}  
width={1000} height={500} data={generateChartData}>  
  <CartesianGrid strokeDasharray="3 3" />  
  <XAxis dataKey="label" />  
  <YAxis  
    domain={mode === 'score' ? [0, 100] : undefined}
```

---

## Tips.jsx

Mostra **consigli personalizzati** in base al punteggio:

- Se il punteggio è basso, consiglia di migliorare la routine.
  - Se è buono, dà suggerimenti per mantenere buone abitudini.
-

## Librerie utilizzate e per farla funzionare..

- **React** – per la costruzione dell'interfaccia.
- **Recharts** – per visualizzazioni grafiche (grafici a torta, linee, barre).
- **Papaparse** – per il parsing dei file CSV nel browser.

Per eseguire correttamente l'applicazione è necessario avere Node.js installato sul proprio PC, insieme alle librerie gestite tramite npm. Una volta configurato l'ambiente, si dovranno eseguire i seguenti comandi da terminale:

```
npm install recharts
```

```
npm install papaparse
```

```
npm start
```

Dopo l'avvio, l'app sarà disponibile in locale (localhost) e potrà essere utilizzata da altri dispositivi connessi alla stessa rete, semplicemente accedendo da browser all'indirizzo IP della macchina host seguito dalla porta (di default 3000).

---

## Conclusione

L'applicazione *Sleep Up* è ben strutturata e facile da estendere.

Utilizza componenti riutilizzabili, una logica chiara, dati e librerie aggiornate per la visualizzazione.

Offre un'interfaccia utente intuitiva che permette analisi interessanti delle abitudini di sonno.

**5/05/2025**

**Indirizzo informatica:**

**Mercede Alessandro, VR504729**

**De Togni Sofia, VR501921**