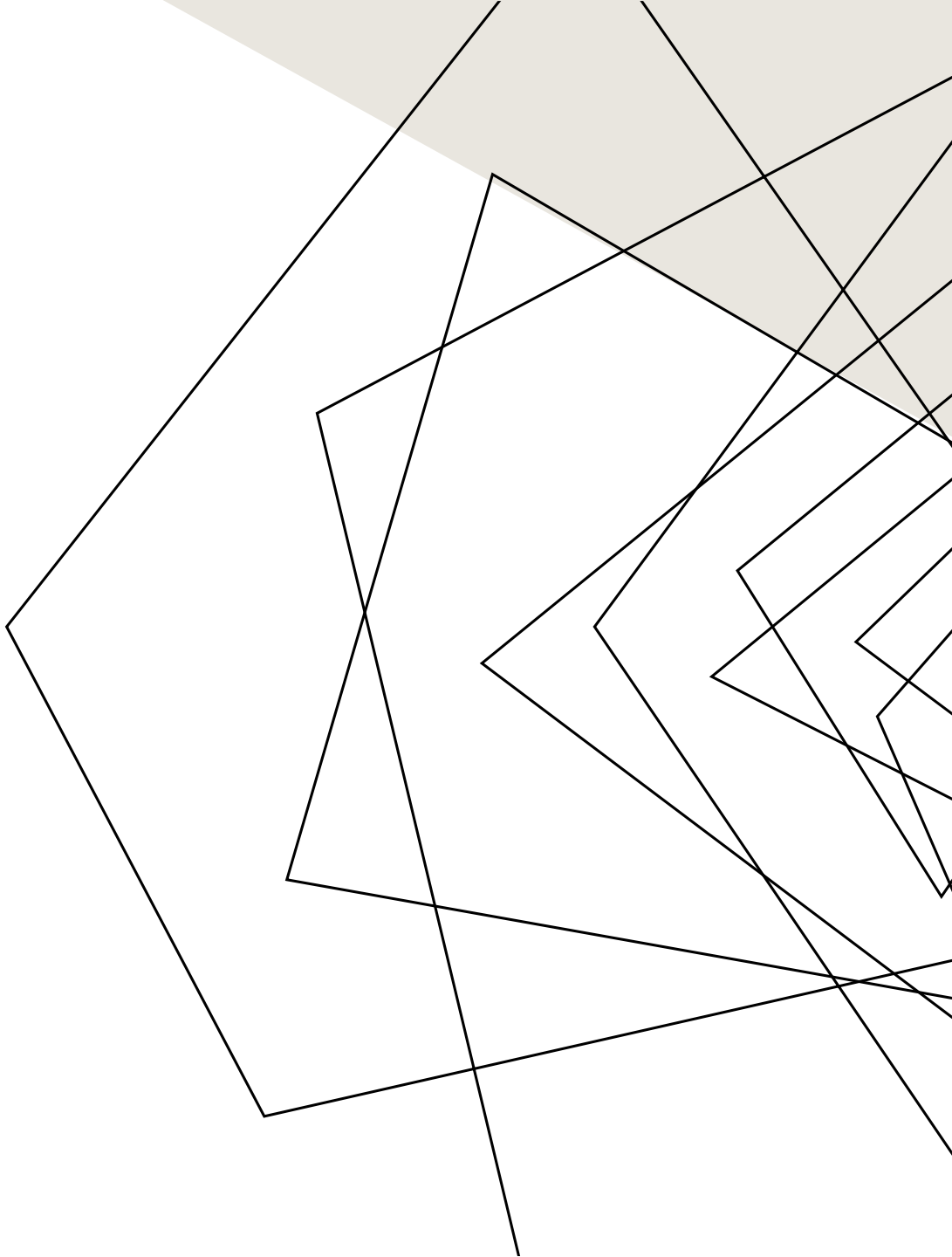
WELLNESS
CENTER

ABOUT US

Simply a group of college students creating things to change the world.

Team Members:

- Brendan
- Isaac
- Dhruvanshi
- Mikayla



WEBSITE OVERVIEW

The website allows users to motivate themselves to be the best version of themself.

As Humber says, "The you, you knew was in you".

WEBSITE BENEFITS

- Improved mindfulness
- Awareness of your physical, mental, and spiritual health
- Actionable tools to help you take next steps in your life

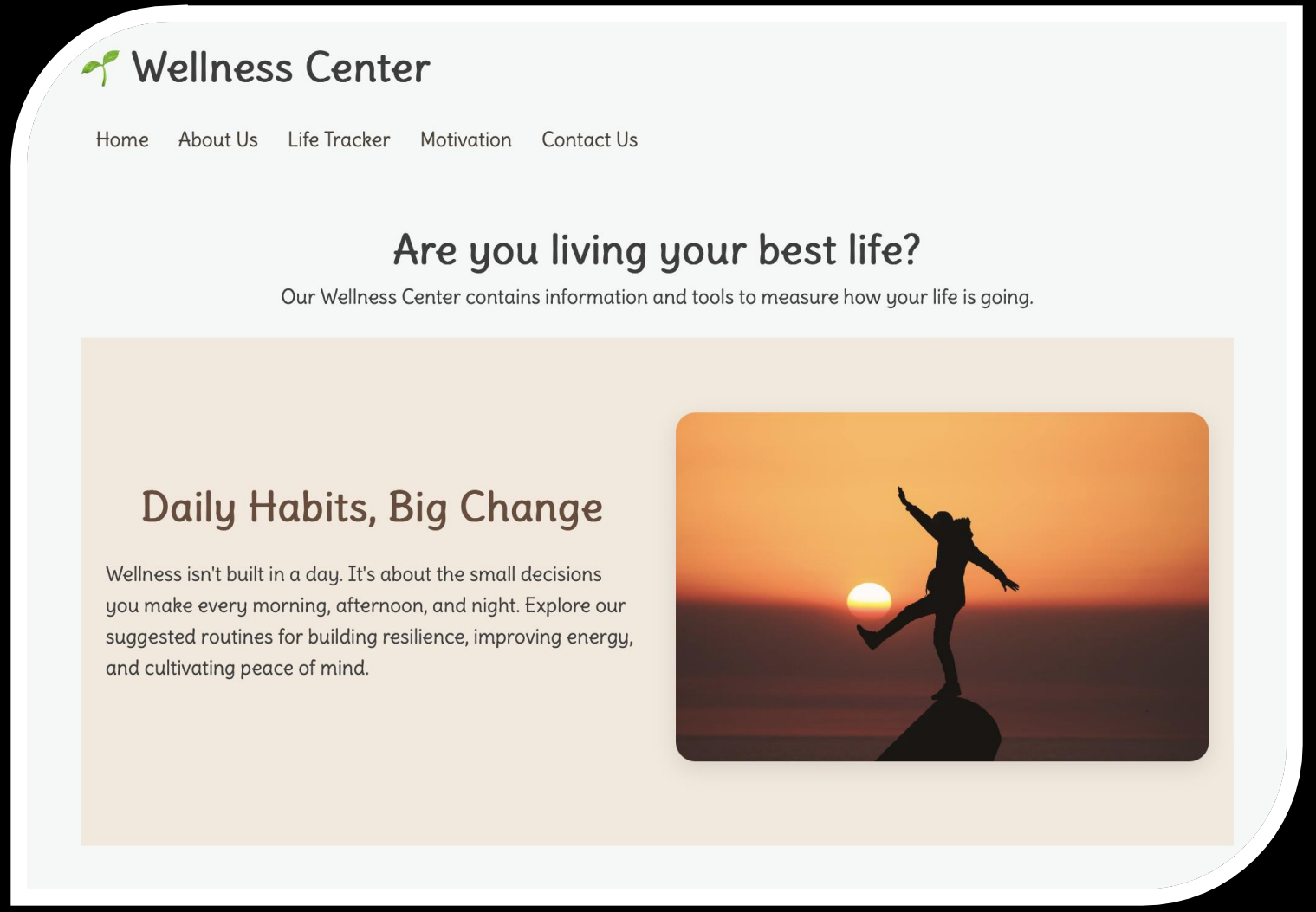
PRODUCT OVERVIEW

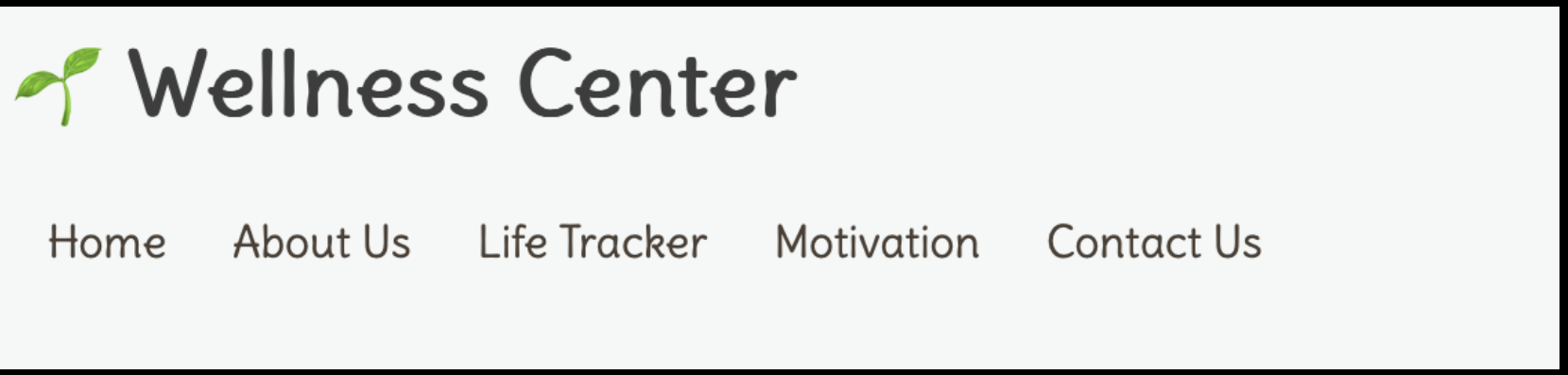
The Wellness Center began with a mission to support our community through a well-rounded, whole-person approach to health and wellness.

- A Life Tracker to help you monitor habits and goals
- Tools for self-care, mindfulness, and personal growth
- quotes to keep you inspired

HOME PAGE

The home page introduces the site and its purpose, and features actionable buttons that link to other parts of the site.



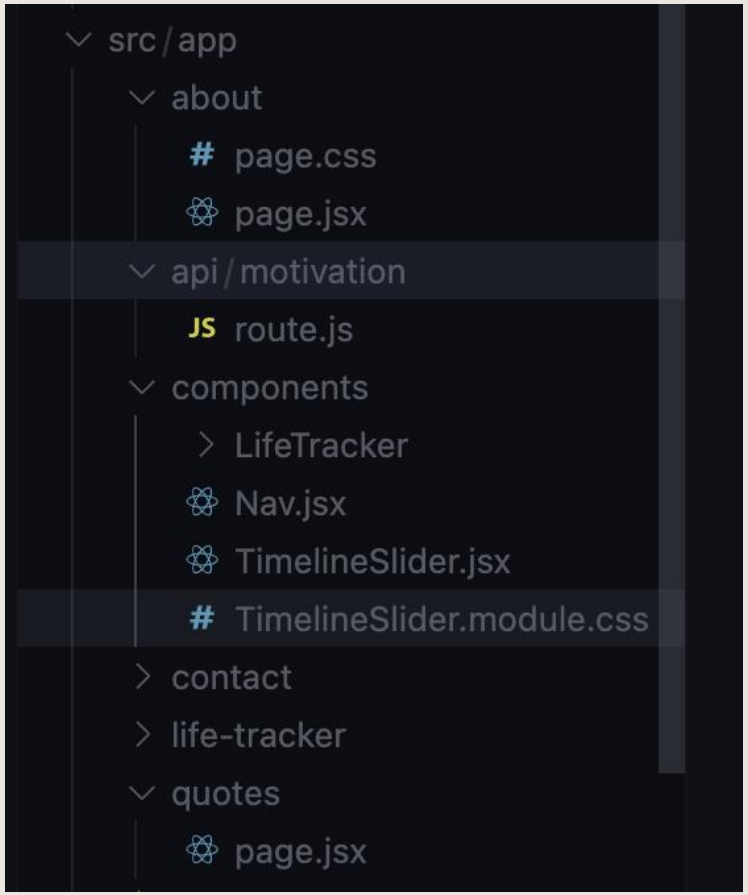


COMMON COMPONENT

The **<Nav />** component is present on every page and reused throughout the site. It contains <Link /> elements to all of our different routes.

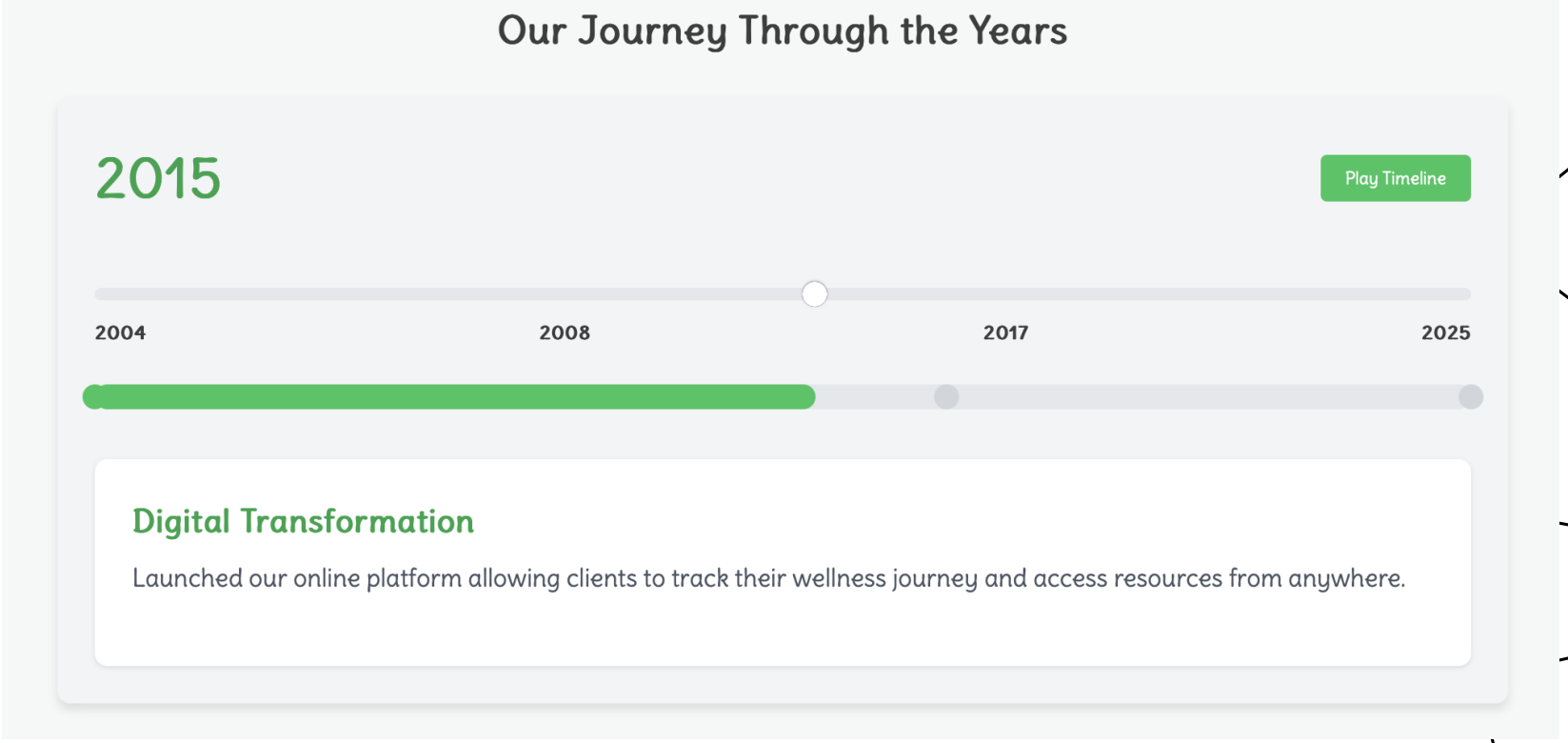
ROUTING

- Routing is implemented using Next.js **page.jsx** files.
- Each route has its own folder under the app directory with a **page.jsx** file inside.
- When navigating using the <Link /> component, the component inside the **page.jsx** file is displayed.



ABOUT PAGE

The about page features an interactive slider component that gives a brief history of the company over the years.



ABOUT PAGE TIMELINE

The component takes advantage of React's ***useState*** function to keep track of the current year on the slider, and event listeners to react to the changes.

src > app > components > TimelineSlider.jsx > TimelineSlider

```jsx
export default function TimelineSlider() {
    const [year, setYear] = useState(2004);
    const [isAnimating, setIsAnimating] = useState(false);

    const milestones = [
        {
            year: 2004,
            title: "Our Beginning",
            description:
                "The Wellness Center was founded with a vision to create a h
                our community.",
        },
        {…
```

```jsx
        <button
            onClick={animateTimeline}
            disabled={isAnimating}
            className={styles.playButton}
        >
            {isAnimating ? "Playing..." : "Play Timeline"}
        </button>
    </div>

    <div className={styles.sliderContainer}>
        <input
            type="range"
            min="2004"
            max="2025"
            value={year}
            onChange={handleSliderChange}
            className={styles.slider}
        />
```

The Play button is also conditionally disabled if the slider animation is in progress.

TIMELINE STYLES

For styling the component uses CSS Modules to keep styles scoped directly to the component.

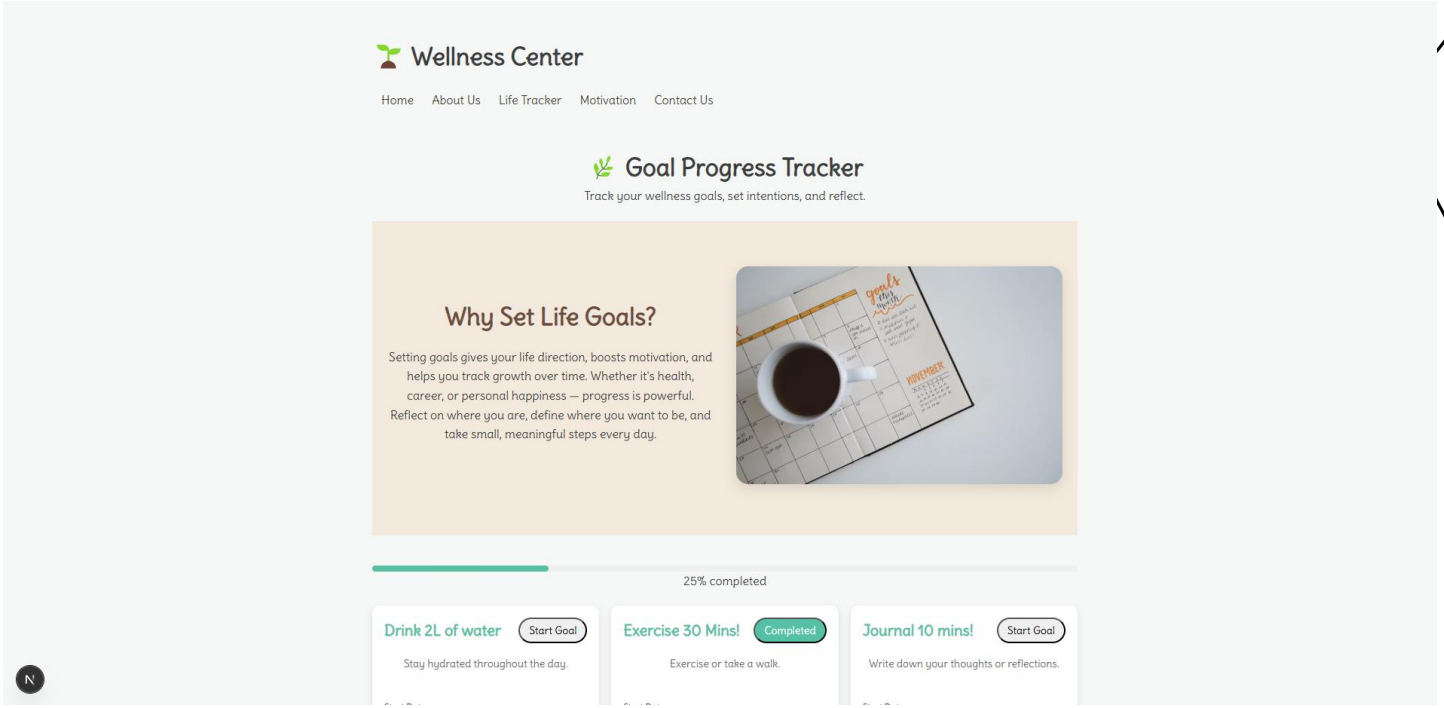
src > app > components > # TimelineSlider.module.css > .progressFill

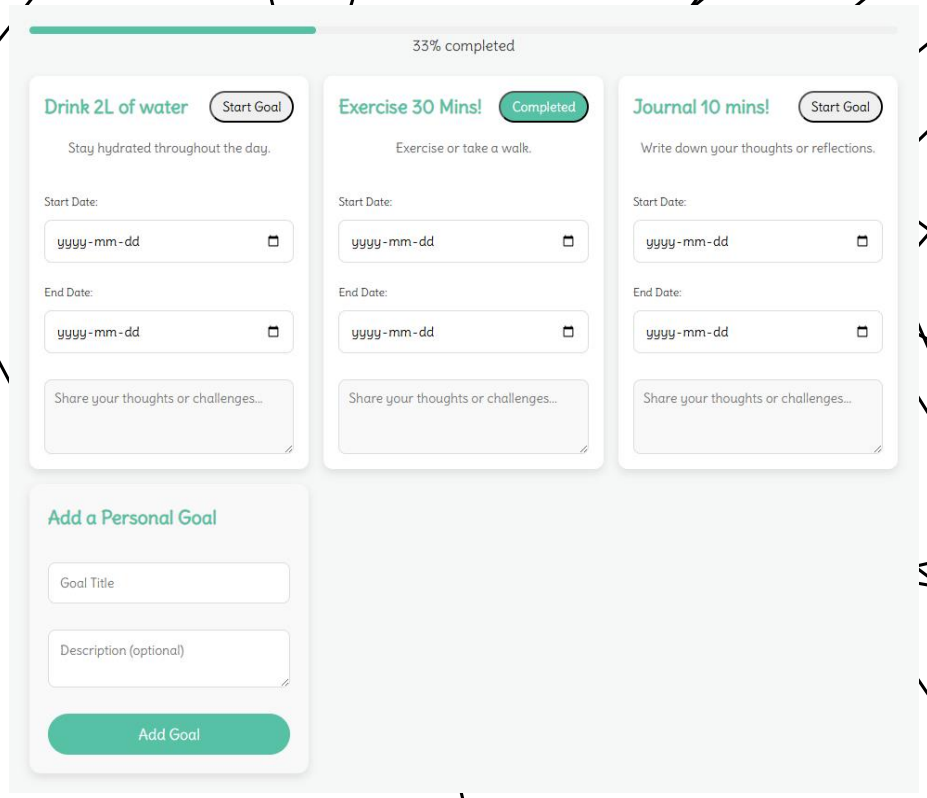
```css
.timelineSection {
    margin-top: 4rem;
    margin-bottom: 4rem;
}

.sectionTitle {
    font-size: 1.5rem;
    font-weight: 700;
    text-align: center;
    margin-bottom: 1.5rem;
}

.timelineContainer {
    background-color: #f3f4f6;
    border-radius: 0.5rem;
    padding: 1.5rem;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}
```

```css
.slider {
    width: 100%;
    height: 0.5rem;
    background-color: #e5e7eb;
    border-radius: 0.5rem;
    appearance: none;
    cursor: pointer;
}

.yearMarkers {
    display: flex;
    justify-content: space-between;
    margin-top: 0.25rem;
}

.yearLabel {
    font-size: 0.75rem;
    font-weight: 600;
}
```

LIFE TRACKER PAGE

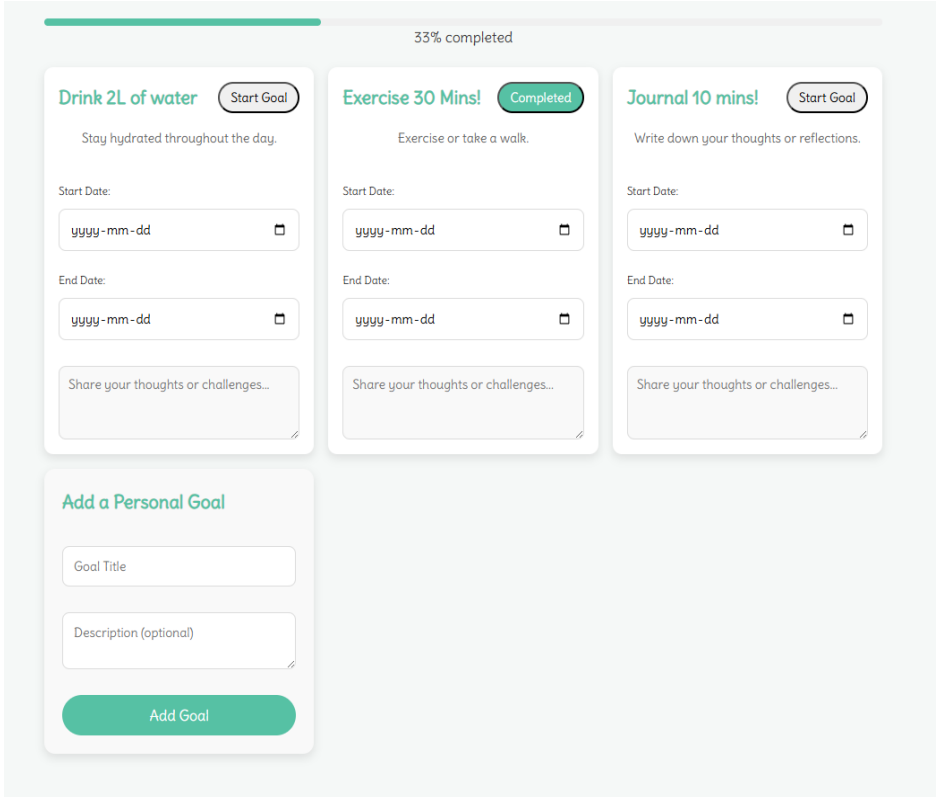
This page features an interactive Goal Progress tracker that helps you track your growth over time. Weather it's health, career, or personal happiness progress is powerful and beautiful.





LIFE TRACKER – GOAL TRACKER COMPONENT

The component takes advantage of React's useState function to keep track of the current goal, start-date/end-date & when completed.

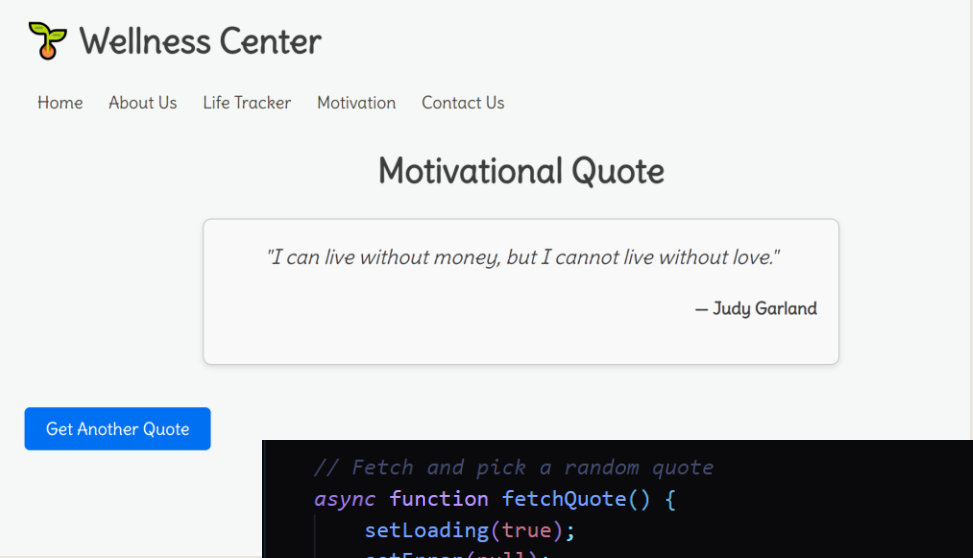



```jsx
import React, { useState } from "react";

export default function GoalCard({ goal, onStart, onComplete }) {
    const [note, setNote] = useState("");
    const [startDate, setStartDate] = useState("");
    const [endDate, setEndDate] = useState("");
```

```jsx
return (
    <div className="goal-card">
        <div className="goal-header ">
            <h2 className="goal-title">{goal.title}</h2>

            {!goal.started && !goal.completed && (
                <button
                    onClick={() => onStart(goal.id)}
                    className="goal-status not-completed"
                >
                    Start Goal
                </button>
            )}

            {goal.started && !goal.completed && (
                <div className="flex">
                    <button className="goal-status not-completed" disabled>
                        In Progress
                    </button>
                    <button
                        onClick={() => onComplete(goal.id)}
                        className="goal-status completed"
                    >
                        Complete
                    </button>
                </div>
            )}

            {goal.completed && (
                <button className="goal-status completed">Completed</button>
            )}
        </div>
```

```jsx
            <p className="goal-description">{goal.description}</p>

            <div className="date-inputs">
                <label className="label">Start Date:</label>
                <input
                    type="date"
                    value={startDate}
                    onChange={(e) => setStartDate(e.target.value)}
                    className="input"
                />

                <label className="label text-xs text-gray-500">End Date:</label>
                <input
                    type="date"
                    value={endDate}
                    onChange={(e) => setEndDate(e.target.value)}
                    className="input"
                />
            </div>

            <textarea
                className="notes"
                placeholder="Share your thoughts or challenges..."
                value={note}
                onChange={(e) => setNote(e.target.value)}
            />
        </div>
    );
```

MOTIVATION PAGE: USAGE OF AN API



This page fetches quotes from https://zenquotes.io/api/random

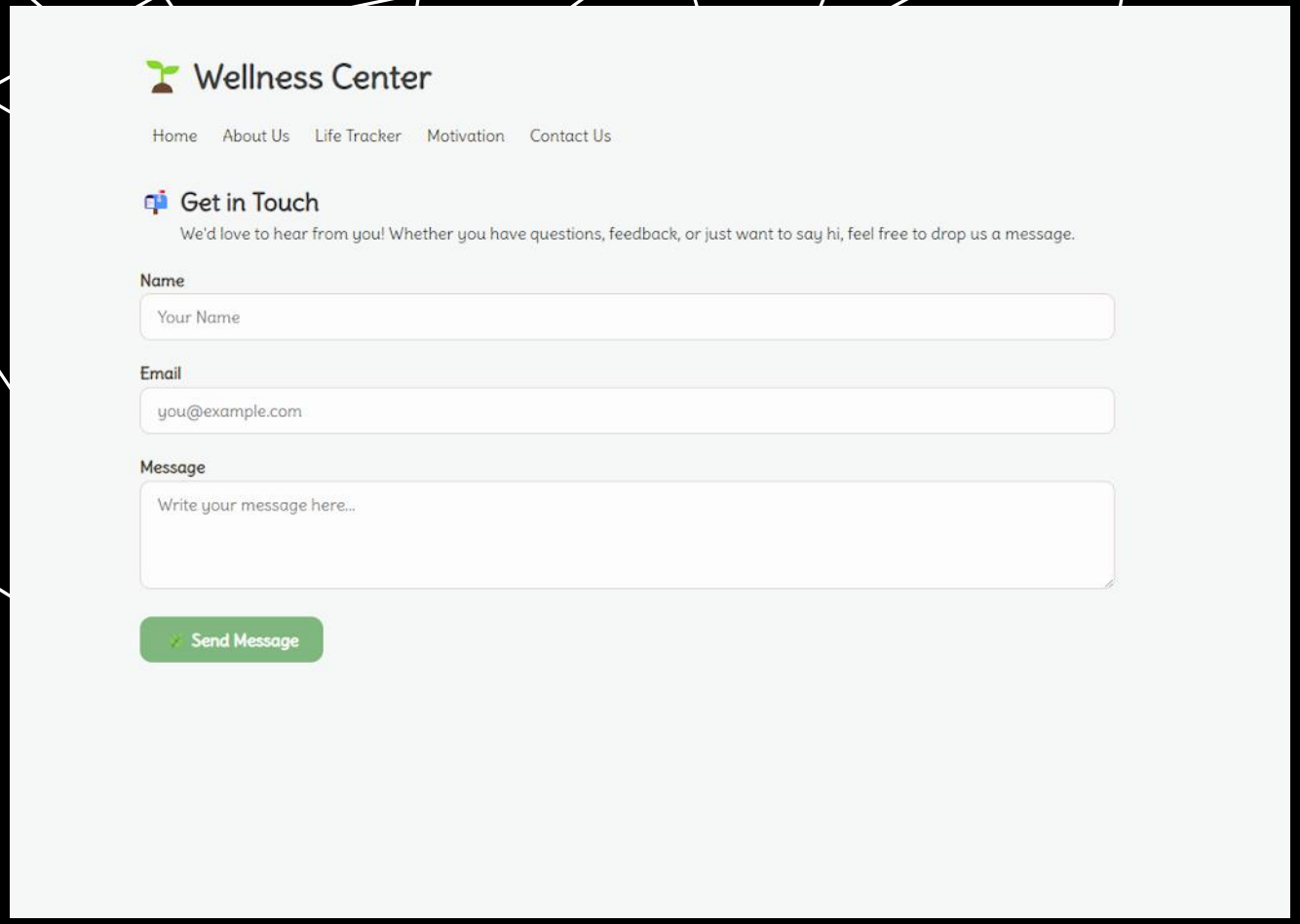
'route.js' is used to bypass CORS (Cross-Origin Resource Sharing) issues, hide API keys, and handle server-side logic. For the quotes page, it serves as a good fallback if the fetching fails.

```
// Fetch and pick a random quote
async function fetchQuote() {
    setLoading(true);
    setError(null);
    try {
        const response = await fetch("/api/motivation");
        if (!response.ok) throw new Error("Failed to fetch quotes");
        const data = await response.json();
        const random = data[Math.floor(Math.random() * data.length)];
        console.log(data, random);
        setQuote(random);
    } catch (err) {
        console.error(err);
        setError("Could not load a quote.");
    } finally {
        setLoading(false);
    }
}
```

route.js

src > app > api > motivation > route.js > GET > res

```
export async function GET() {
    try {
        const res = await fetch("https://zenquotes.io/api/random");
        if (!res.ok) throw new Error("External API error");

        const data = await res.json();
        return Response.json(data);
    } catch (error) {
        return new Response(
            JSON.stringify({ error: "Failed to fetch quote" }),
            {
                status: 500,
            }
        );
    }
}
```



CONTACT PAGE

The Contact page allows users to get in touch by filling out a form or reaching out via provided contact details. It ensures easy communication for inquiries, support, or feedback.

CONTACT PAGE

```
export default function ContactPage() {
  const [formData, setFormData] = useState({
    name: '',
    email: '',
    message: '',
  });
```

- Initializes state to store form input values.
- Keeps track of user input dynamically.

CONTACT PAGE

```
const handleChange = (e) => {
  setFormData({ ...formData, [e.target.name]: e.target.value });
};
```

- Updates the respective field in the state as the user types.
- Uses the name attribute to determine which field is being updated.

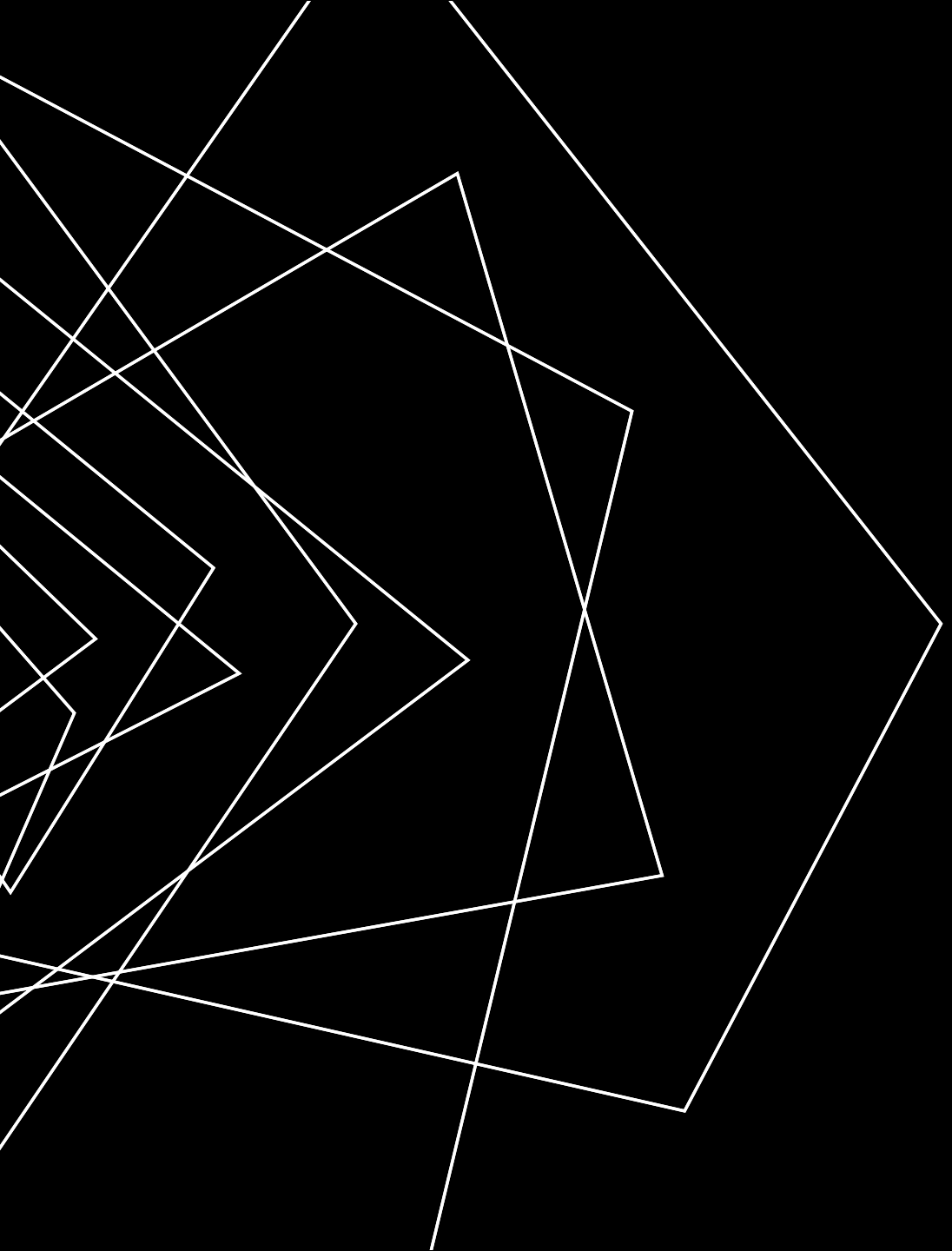
```
const handleSubmit = (e) => {
  e.preventDefault();
  alert('Thank you for your message! We will get back to you soon.');
  setFormData({ name: '', email: '', message: '' });
};
```

- Prevents page reload on submit.
- Shows an alert and resets the form after submission.

CONTACT PAGE

```jsx
<div>
  <label className="block text-[#4b3e2e] font-semibold mb-1">Name</label>
  <input
    type="text"
    name="name"
    className="w-full px-4 py-2 border border-[#ddd5ca] rounded-xl bg-[#fafaf9] focus:outline-none focus:ring-2 focus:ring-green-300"
    placeholder="Your Name"
    value={formData.name}
    onChange={handleChange}
    required
  />
</div>
<div>
  <label className="block text-[#4b3e2e] font-semibold mb-1">Email</label>
  <input
    type="email"
    name="email"
    className="w-full px-4 py-2 border border-[#ddd5ca] rounded-xl bg-[#fafaf9] focus:outline-none focus:ring-2 focus:ring-green-300"
    placeholder="you@example.com"
    value={formData.email}
    onChange={handleChange}
    required
  />
</div>
<div>
  <label className="block text-[#4b3e2e] font-semibold mb-1">Message</label>
  <textarea
    name="message"
    rows="4"
    className="w-full px-4 py-2 border border-[#ddd5ca] rounded-xl bg-[#fafaf9] focus:outline-none focus:ring-2 focus:ring-green-300"
    placeholder="Write your message here..."
    value={formData.message}
    onChange={handleChange}
    required
  />
</div>
```

- Controlled components ensure input values are tied to state.
- Required fields prevent empty submissions.



THANK YOU