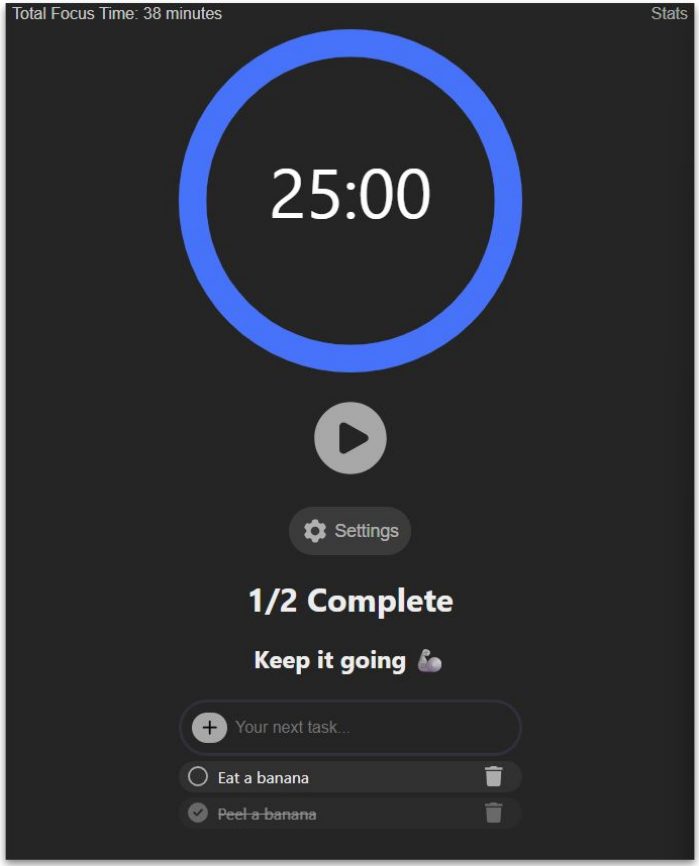
Case Study for Tomato Time Project



Overview

Tomato Time is a responsive Pomodoro timer web application designed to help users stay focused and productive during work or study sessions. It features customizable timers, focus tracking, motivational to-do lists, and a clean, intuitive user interface built with React.

Purpose & Context

This project was created as part of my curiosity towards web technologies such as React.js. Pomodoro Technique helps me to focus, so this was a great choice to learn more.

Objective

The goal of Tomato Time was to build a polished, portfolio-ready application that solves a genuine problem — managing focus and productivity. It highlights my ability to design and develop a user-centered React app from start to finish, incorporating both core functionality and thoughtful user experience design.

Approach

I began by identifying the core functionality of a Pomodoro timer: timed work sessions, breaks, and a user-friendly interface. I prioritized simplicity and responsiveness, aiming for an experience that required minimal user effort to start a session while still offering helpful customization options.

Using React with Hooks, I built the timer logic and state management to support session cycling, countdowns, and break transitions. I implemented LocalStorage to preserve user preferences and to-do list items between visits, ensuring the app could support consistent daily use.

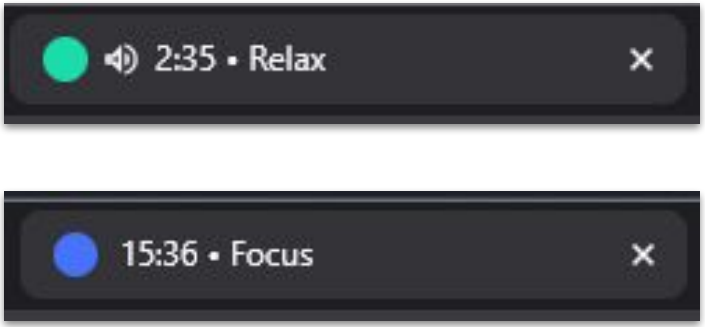
```
function tick() {
  secondsLeftRef.current--;
  setSecondsLeft(secondsLeftRef.current);

  if (modeRef.current === "work") {
    const todayKey = new Date().toISOString().slice(0, 10);
    const focusData = JSON.parse(localStorage.getItem("focusData") || "{}");

    focusData[todayKey] = (focusData[todayKey] || 0) + 1; // +1 second
    localStorage.setItem("focusData", JSON.stringify(focusData));

    // Optional: update totalFocusMinutes too if you use it elsewhere
    const totalFocus = parseInt(
      localStorage.getItem("totalFocusMinutes") || "0",
      10
    );
    localStorage.setItem("totalFocusMinutes", (totalFocus + 1).toString());
  }
}
```

User Interface



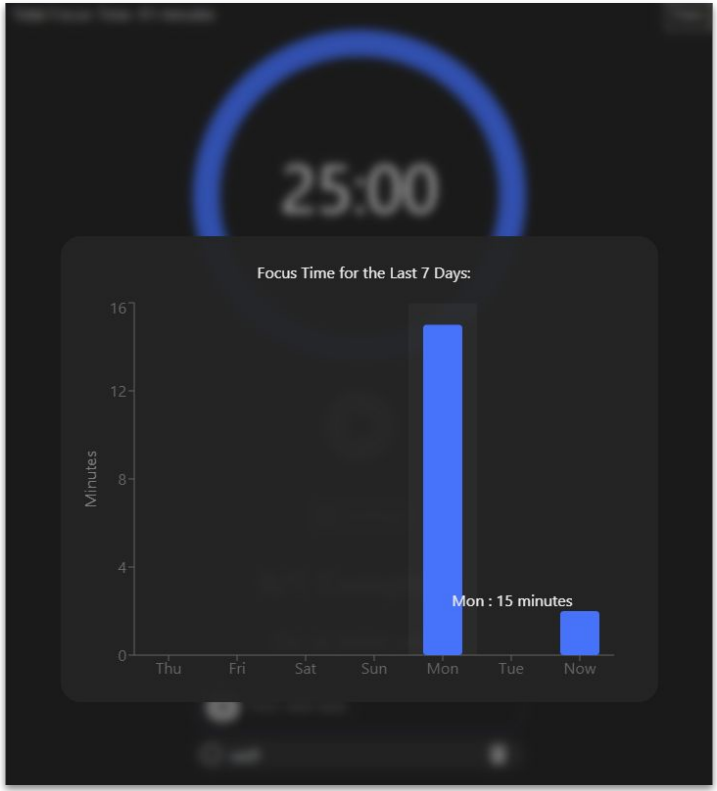
For the UI, I designed an intuitive layout that visually communicates session progress and status. Animations and browser tab updates keep users informed, even when the app isn't in focus. Throughout development, I followed component-based design principles and iteratively tested features to enhance performance and usability.

Challenges

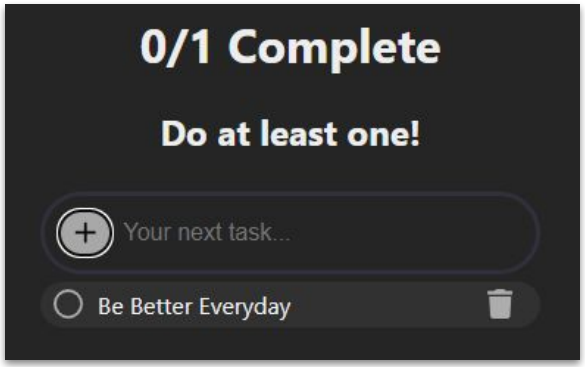
Managing accurate timer transitions using `setInterval` and `useEffect` in React was a key challenge, especially when switching between work and break sessions. Ensuring proper interval cleanup was critical to prevent bugs.

Persisting user data—like to-do items and completed sessions—required syncing React state with LocalStorage. For the 7-day focus chart, I stored session data locally and updated it every minute, which involved careful timestamp tracking and efficient state updates.

Balancing a clean UI with real-time feedback (like sound cues, tab title updates, and animations) pushed me to fine-tune both design and performance across different devices.



Duration



This project took approximately two months to complete, as I was learning React while actively building the app. The timeline reflects the time and effort needed to understand key React concepts and apply them effectively in a real-world project.

Thank You For Viewing!

Would you like to connect with me?

My Gmail: VincentAThorne2005@gmail.com