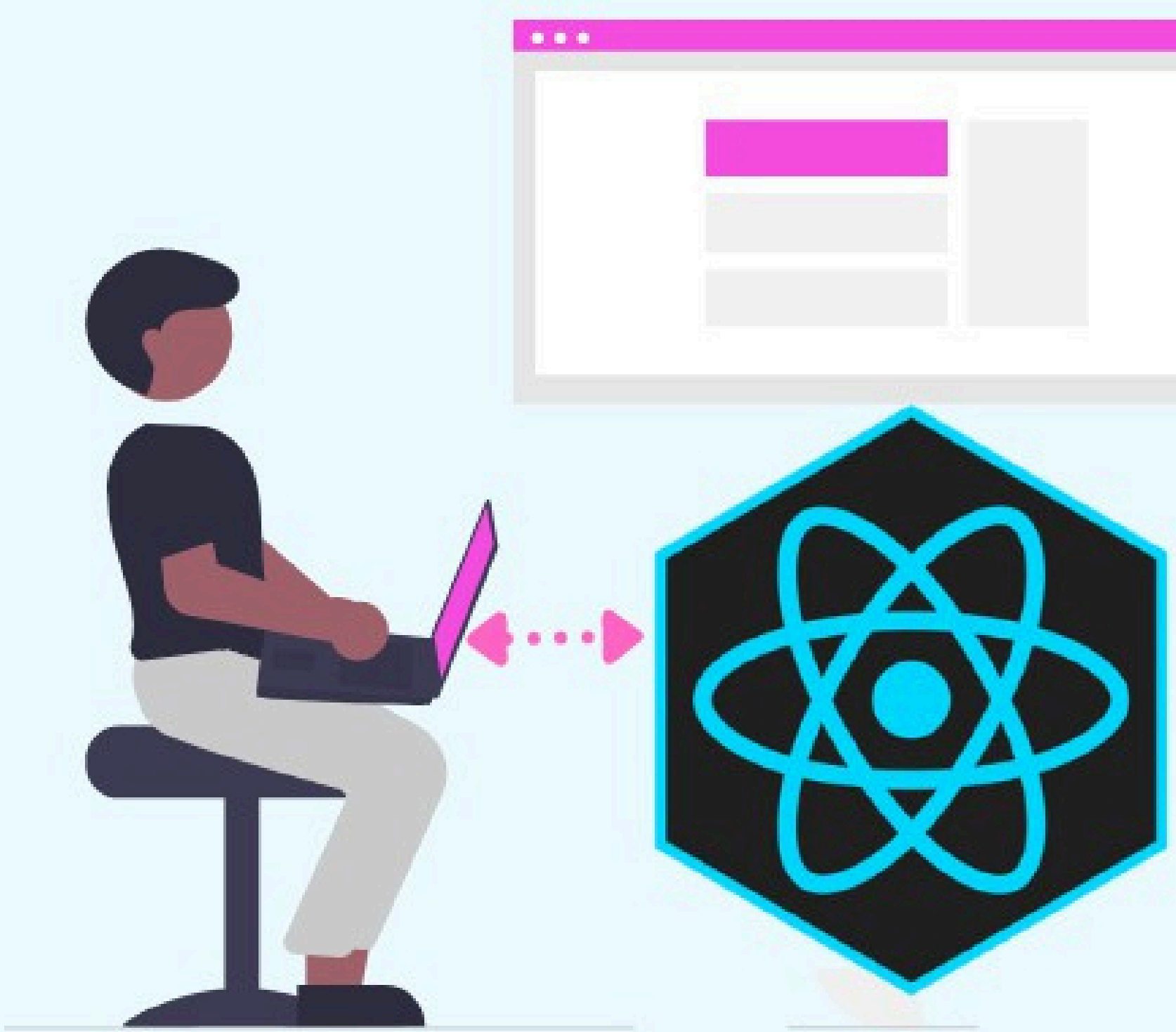
Day 27

Shifting from Theory to Practice (Project Day 3)



@oluwakemi Oluwadahunsi

What we will be doing today?

- Create the MoodStatistics page
- Create the CustomMoodLabels component
- Create the Settings section
- Create the ErrorBoundary component
- Update the App.jsx file, view snapshots of the app
- Push to Github and deploy the app

Things to note:

- **Remember to import and wrap your application with the MoodContext Provider in your main.jsx file like this:**

```
<ThemeProvider>
    <MoodProvider>
      <App />
    </MoodProvider>
  </ThemeProvider>
```

@oluwakemi Oluwadahunsi

- Be sure to use correct paths for imports of components and files. If the files paths are not correct in mine, fix it in yours, else it will throw errors, our project structure may be different.

- In our App.jsx file, we will be updating the newly created components and fix the stylings, be sure to update yours too to this latest styles.

- There would be component codes images, some could be more than 1, I will indicate on each image which component it is together with the image number, just continue in the same component according to their numbers.

Mood Statistics Page

This is the page rendered when the “**Statistics**” link is clicked on the header of the application.

It gives insights using charts to visualize and calculate the logged mood by percentages and also the average of the moods logged for a particular day. We will be using the “recharts” library for this.

The MoodStatistics component makes use of a static component named “Chart.jsx” where one of the charts is being styled to avoid overloading a single component.

First, let’s create the static component “**Chart.jsx**” in the **staticComponents** folder:

Chart.jsx image 1

```jsx
import React from "react";
import { ResponsiveContainer, PieChart, Pie, Cell, Tooltip, BarChart, Bar,
XAxis, YAxis, CartesianGrid, Legend } from "recharts";

const COLORS = [
  "#0088FE", "#00C49F", "#FFBB28", "#FF8042", "#8884D8", "#82ca9d", "#ffc658",
];

const Chart = ({ data, type }) => {
  if (type === "pie") {
    return (
      <ResponsiveContainer width="100%" height={300}>
        <PieChart>
          <Pie
            data={data}
            dataKey="count"
            nameKey="mood"
            cx="50%"
            cy="50%"
            outerRadius={100}
            fill="#8884d8"
            label={({ name, percent }) =>
              `${name} ${(percent * 100).toFixed(0)}%`
            }
          >
            {data.map((entry, index) => (
              <Cell
                key={`cell-${index}`}
                fill={COLORS[index % COLORS.length]}
              />
            ))}
          </Pie>
          <Tooltip />
          <Legend />
        </PieChart>
      </ResponsiveContainer>
    );
  }
```

Chart.jsx image 2

```jsx
else if (type === "bar") {
    return (
      <ResponsiveContainer width="100%" height={300}>
        <BarChart
          data={data}
          margin={{ top: 20, right: 30, left: 20, bottom: 5 }}
        >
          <CartesianGrid strokeDasharray="3 3" />
          <XAxis dataKey="date" />
          <YAxis />
          <Tooltip />
          <Legend />
          <Bar dataKey="averageMood" fill="#8884d8" name="Average Mood" />
        </BarChart>
      </ResponsiveContainer>
    );
  }
  return null;
};

export default React.memo(Chart);
```

Now, create a new file named “**MoodStatistics.jsx**” in your components folder:

@oluwakemi Oluwadahunsi

MoodStatistics image 1

```jsx
import React, { useContext, useMemo } from "react";
import { MoodContext } from "../hooks/MoodContext";
import { endOfWeek, format, startOfWeek, eachDayOfInterval } from "date-fns";
import Chart from "./pages/staticPages/Chart";

const moods = [
  { emoji: "😄", label: "Happy", value: 7 },
  { emoji: "🎉", label: "Excited", value: 6 },
  { emoji: "😊", label: "Calm", value: 5 },
  { emoji: "😌", label: "Relaxed", value: 4 },
  { emoji: "😐", label: "Neutral", value: 3 },
  { emoji: "😔", label: "Sad", value: 2 },
  { emoji: "😠", label: "Angry", value: 1 },
];

function MoodStatistics() {
  const { moodEntries } = useContext(MoodContext);

  const moodStats = useMemo(() => {
    const stats = {};
    moodEntries.forEach((entry) => {
      if (entry && entry.mood) {
        const label = entry.mood.label;
        if (stats[label]) {
          stats[label]++;
        } else {
          stats[label] = 1;
        }
      }
    });
    return Object.entries(stats).map(([mood, count]) => ({ mood, count }));
  }, [moodEntries]);

  const mostFrequentMood = useMemo(() => {
    return moodStats.reduce((a, b) => (a.count > b.count ? a : b), {
      mood: "",
      count: 0,
    });
  }, [moodStats]);

  const weeklyMoodTrend = useMemo(() => {
    const today = new Date();
    const startOfWeekDate = startOfWeek(today);
    const endOfWeekDate = endOfWeek(today);

    const daysOfWeek = eachDayOfInterval({
      start: startOfWeekDate,
      end: endOfWeekDate,
    });
```

@oluwakemi Oluwadahunsi

MoodStatistics image 2

```
return daysOfWeek.map((day) => {
      const dayStr = format(day, "EEE");
      const dayMoods = moodEntries.filter((entry) => {
        const entryDate = new Date(entry.date);
        return format(entryDate, "EEE") === dayStr;
      });

      const moodValues = dayMoods
        .map((entry) => {
          const mood = moods.find((m) => m.label === entry.mood.label);
          return mood ? mood.value : 0;
        })
        .filter((value) => value !== 0);

      const averageMood =
        moodValues.length > 0
          ? moodValues.reduce((sum, value) => sum + value, 0) /
            moodValues.length
          : null;

      return {
        date: dayStr,
        averageMood:
          averageMood !== null ? Number(averageMood.toFixed(2)) : null,
      };
    });
  }, [moodEntries]);

  if (moodEntries.length === 0) {
    return (
      <div className="text-center p-4">
        <p className="text-lg text-gray-600 dark:text-gray-300">
          No mood data available yet. Start logging your moods to see
          statistics!
        </p>
      </div>
    );
  }

  return (
    <div className="mb-8">
      <h2 className="text-lg sm:text-2xl font-bold mb-4 text-gray-800
dark:text-white">
        Mood Statistics
      </h2>
```

@oluwakemi Oluwadahunsi

MoodStatistics image 3

```jsx
<div className="mb-6">
        <h3 className="text-base sm:text-xl font-semibold mb-2 text-gray-700 dark:text-gray-300">
          Most Frequent Mood
        </h3>
        <p className="text-sm sm:text-lg  text-gray-700 dark:text-gray-300">
          {mostFrequentMood.mood} ({mostFrequentMood.count} times)
        </p>
      </div>

      {moodStats.length > 0 && (
        <div className="mb-6">
          <h3 className="text-base sm:text-xl font-semibold mb-2 text-gray-700 dark:text-gray-300">
            Mood Distribution
          </h3>
          <Chart data={moodStats} type="pie" />
        </div>
      )}

      {weeklyMoodTrend.length > 0 && (
        <div className="mb-6">
          <h3 className="text-base sm:text-xl font-semibold mb-2 text-gray-700 dark:text-gray-300">
            Weekly Mood Trend
          </h3>
          <Chart data={weeklyMoodTrend} type="bar" />
        </div>
      )}
    </div>
  );
}

export default React.memo(MoodStatistics);
```

@oluwakemi Oluwadahunsi

CustomMoodLabels component

This component is for allowing users to create their custom labels in cases where the default labels available does not reflect their current moods.

This component makes use of a component named "**EmojiChart.jsx**" where we styled the UI for the emojis. This EmojiChart component also uses another file named "**emojis.js**", where we'll store the emojis.

I will be giving a few emojis for description purpose, you can add as many as you want to each category.

Now, create a new file in the **staticComponents** folder named **emojis.js**

@oluwakemi Oluwadahunsi

emojis.js

```
export const emojiCategories = [
  {
    name: "Smileys & Emotion",
    emojis: [
      "😀", "🙃", "😉", "😊", "😇", "🥰", "😍", "🤩", "😘",
    ],
  },
  {
    name: "People & Body",
    emojis: [
      "🤦‍♀️", "🤦‍♂️", "🤷‍♀️", "🤷‍♂️", "💃", "🕺","🧘‍♂️", "👋", "🤚", "🖐️", "✋", "🤙",
      "👈", "👉", "🤲", "🤝", "🙏",
    ],
  },
  {
    name: "Animals & Nature",
    emojis: [
      "🐶", "🐱", "🐭", "🐹", "🐰", "🦊", "🐻", "🐼", "🐨", "🐯", "🦁", "🐮",
    ],
  },
];
```

Create a file named **EmojiChart.jsx** in your **staticComponents** folder:

Then in your **components** folder, create a new file named “**CustomMoodLabels.jsx**”.

@oluwakemi Oluwadahunsi

EmojiChart.jsx

```jsx
import { X } from "lucide-react";
import { emojiCategories } from "./emojis";

export const EmojiChart = ({ onSelect, onClose }) => {
  return (
    <div className="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center z-50">
      <div className="bg-white dark:bg-gray-800 p-4 rounded-lg max-w-lg w-full max-h-[80vh] overflow-y-auto relative">
        <h3 className="text-lg font-semibold mb-4 text-gray-900 dark:text-white">
          Select an Emoji
        </h3>
        <div
          onClick={onClose}
          className="absolute top-5 right-10 cursor-pointer text-gray-900 dark:text-white"
        >
          <X />
        </div>

        {emojiCategories?.map((category) => (
          <div key={category.name} className="mb-4">
            <h4 className="text-md font-medium mb-2 text-gray-700 dark:text-gray-300">
              {category.name}
            </h4>
            <div className="grid grid-cols-8 gap-2">
              {category.emojis.map((emoji) => (
                <button
                  key={emoji}
                  onClick={() => {
                    onSelect(emoji);
                    onClose();
                  }}
                  className="text-2xl hover:bg-gray-200 dark:hover:bg-gray-700 rounded p-1"
                >
                  {emoji}
                </button>
              ))}
            </div>
          </div>
        ))}
        <button
          onClick={onClose}
          className="mt-4 w-full bg-red-500 text-white py-2 px-4 rounded-md hover:bg-red-600 transition-colors duration-200"
        >
          Close
        </button>
      </div>
    </div>
  );
};
```

@oluwakemi Oluwadahunsi

CustomMoodLabels image 1

```jsx
import { useState, useContext } from "react";
import { MoodContext } from "../hooks/MoodContext";
import { EmojiChart } from "./pages/staticPages/EmojiChart";

function CustomMoodLabels() {
  const moodContext = useContext(MoodContext);
  const [showEmojiChart, setShowEmojiChart] = useState(false);
  const [newMood, setNewMood] = useState({ label: "", emoji: "", color: "" });

  if (!moodContext) {
    console.error("MoodContext is not available");
    return <div>Error: MoodContext is not available</div>;
  }

  const { customMoods, addCustomMood } = moodContext;

  const handleEmojiSelect = (emoji) => {
    setNewMood((prev) => ({ ...prev, emoji }));
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    if (newMood.label && newMood.emoji && newMood.color) {
      if (typeof addCustomMood === "function") {
        addCustomMood(newMood);
        setNewMood({ label: "", emoji: "", color: "" });
      } else {
        console.error("addCustomMood is not a function");
      }
    }
  };

  return (
    <div className="mb-8">
      <form onSubmit={handleSubmit} className="mb-4 w-full">
        <h3 className="text-lg sm:text-2xl font-semibold mb-2 text-gray-800 dark:text-white">
          Add Custom Mood
        </h3>
        <div className="mb-2">
          <label
            htmlFor="moodLabel"
            className="block text-sm font-medium text-gray-700 dark:text-gray-300"
          >
            Mood Label
          </label>
          <input
            type="text"
            id="moodLabel"
            value={newMood.label}
            onChange={(e) =>
              setNewMood((prev) => ({ ...prev, label: e.target.value }))
            }
            className="p-2 border rounded-md dark:bg-gray-700 dark:text-white w-full"
            required
          />
        </div>
```

CustomMoodLabels image 2

```jsx
<div className="w-full flex justify-between">
          <div className="mb-2 w-full">
            <label
              htmlFor="moodEmoji"
              className="block text-sm font-medium text-gray-700 dark:text-gray-300"
            >
              Mood Emoji
            </label>
            <div className="flex items-center">
              <input
                type="text"
                id="moodEmoji"
                value={newMood.emoji}
                onChange={(e) =>
                  setNewMood((prev) => ({ ...prev, emoji: e.target.value }))
                }
                className="w-1/2 sm:w-3/4 p-2 border rounded-md dark:bg-gray-700 dark:text-white"
                required
                readOnly
              />
              <button
                type="button"
                onClick={() => setShowEmojiChart(true)}
                className="w-1/2 sm:w-1/4 ml-2 bg-indigo-600 text-white text-sm sm:text-base py-3 sm:py-2 px-2 sm:px-4 rounded-md hover:bg-indigo-700 transition-colors duration-200"
              >
                Choose Emoji
              </button>
            </div>
          </div>
        </div>
        <div className="mb-2">
          <label
            htmlFor="moodColor"
            className="block text-sm font-medium text-gray-700 dark:text-gray-300"
          >
            Mood Color (e.g., #FF0000 for red)
          </label>
          <input
            type="color"
            id="moodColor"
            value={newMood.color}
            onChange={(e) =>
              setNewMood((prev) => ({ ...prev, color: e.target.value }))
            }
            className="w-full p-1 border rounded-md dark:bg-gray-700"
            required
          />
        </div>
        <button
          type="submit"
          className="w-full bg-green-500 text-white py-2 px-4 rounded-md hover:bg-green-600 transition-colors duration-200"
        >
          Add Custom Mood
        </button>
      </form>

```

@oluwakemi Oluwadahunsi

CustomMoodLabels image 3

```
<div className="grid grid-cols-2 sm:grid-cols-3 gap-4">
        {Array.isArray(customMoods) && customMoods.length > 0 ? (
          customMoods.map((mood, index) => (
            <div
              key={index}
              className="p-4 rounded-lg text-center"
              style={{ backgroundColor: mood.color }}
            >
              <span className="text-3xl" role="img" aria-label={mood.label}>
                {mood.emoji}
              </span>
              <p className="mt-2 text-sm font-medium">{mood.label}</p>
            </div>
          ))
        ) : (
          <p className="col-span-full text-center text-gray-500 dark:text-gray-
400">
            No custom moods added yet.
          </p>
        )}
      </div>

      {showEmojiChart && (
        <EmojiChart
          onSelect={handleEmojiSelect}
          onClose={() => setShowEmojiChart(false)}
        />
      )}
    </div>
  );
}

export default CustomMoodLabels;
```

@oluwakemi Oluwadahunsi

ErrorBoundary Component

An Error Boundary in React is a **special type** of component used to catch **JavaScript errors** that occur anywhere in the **child component** tree and gracefully handle them.

It helps prevent the entire React application from crashing due to a single component's error.

Create the **ErrorBoundary.jsx** component in the **components** folder.

@oluwakemi Oluwadahunsi

ErrorBoundary.jsx

```jsx
import React from "react";

class ErrorBoundary extends React.Component {
  constructor(props) {
    super(props);
    this.state = { hasError: false };
  }

  static getDerivedStateFromError() {
    return { hasError: true };
  }

  componentDidCatch(error, errorInfo) {
    console.error("Error caught by ErrorBoundary:", error, errorInfo);
  }

  render() {
    if (this.state.hasError) {
      return (
        <div className="min-h-screen flex items-center justify-center bg-red-100 dark:bg-red-900">
          <div className="text-center">
            <h1 className="text-3xl font-bold text-red-800 dark:text-red-200 mb-4">
              Oops! Something went wrong.
            </h1>
            <p className="text-lg text-red-600 dark:text-red-300">
              We&apos;re sorry for the inconvenience. Please check the developer console for details.
            </p>
          </div>
        </div>
      );
    }

    return this.props.children;
  }
}

export default ErrorBoundary;
```

displays a message when an error is caught

@oluwakemi Oluwadahunsi

Updating the App.jsx

Let’s update our App.jsx with the newly created components, and also watch out for the updated stylings for the application.

Major changes:

- Newly created components are included.
- More active states for header are added for pages navigation.
- UI styles are updated
- A message toast component from “**sonner**” is added for displaying messages. To install this library, run “**npm install sonner**” in your terminal.

Check out how the toast is being used in the MoodSelector, RecentEntries, and CustomMoodLabels components, in the **GitHub repository** I will be sharing later.

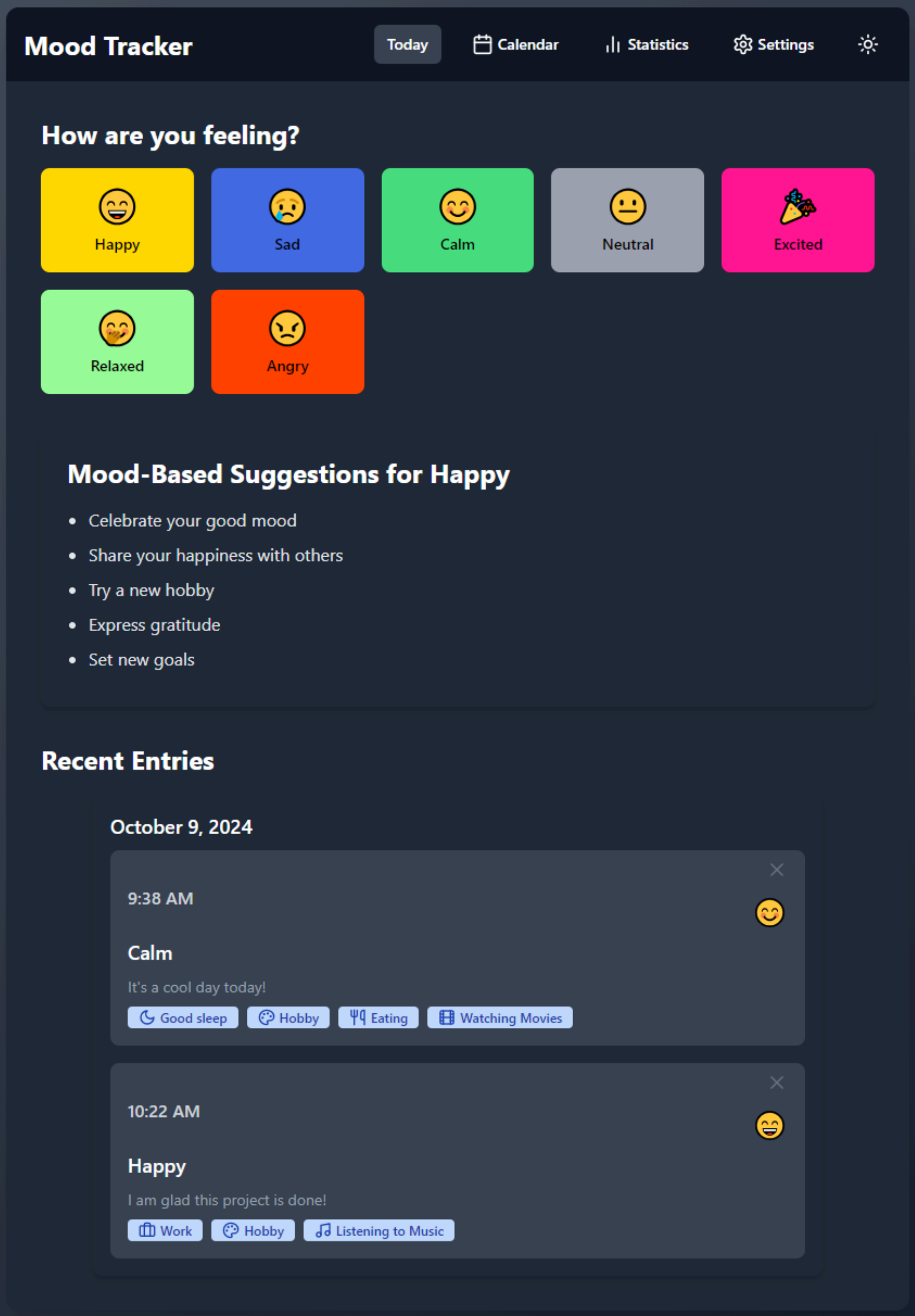
@oluwakemi Oluwadahunsi

App.jsx

```jsx
import { lazy, Suspense, useState } from "react";
import Header from "./components/Header";
import LoadingFallback from "./components/staticComponents/LaodingFallback";
import ErrorBoundary from "./components/ErrorBoundary";
import { Toaster } from "sonner";

const MoodSelector = lazy(() => import("./components/MoodSelector"));
const MoodCalendar = lazy(() => import("./components/MoodCalendar"));
const MoodStatistics = lazy(() => import("./components/MoodStatistics"));
const ActivityTracker = lazy(() => import("./components/ActivityTracker"));
const RecentEntries = lazy(() => import("./components/RecentEntries"));
const CustomMoodLabels = lazy(() => import("./components/CustomMoodLabels"));
const MoodSuggestions = lazy(() => import("./components/MoodSuggestions"));

function App() {
  const [activeTab, setActiveTab] = useState("today");

  return (
    <ErrorBoundary>
      <div className="min-h-screen bg-gradient-to-br from-purple-400 to-indigo-600 dark:from-gray-600 dark:to-gray-900 transition-colors duration-200 shadow-2xl">
        <div className="max-w-4xl mx-auto px-4 sm:px-6 lg:px-8 py-6 sm:py-8 lg:py-12">
          <div className="bg-white dark:bg-gray-800 rounded-xl shadow-xl overflow-hidden transition-colors duration-200">
            <Toaster position="top-right" />
            <Header activeTab={activeTab} setActiveTab={setActiveTab} />
            <main className="p-4 sm:p-6 lg:p-8">
              <Suspense fallback={<LoadingFallback />}>
                {activeTab === "today" && (
                  <div className="space-y-6 sm:space-y-8">
                    <MoodSelector />
                    <ActivityTracker />
                    <MoodSuggestions />
                    <RecentEntries />
                  </div>
                )}
                {activeTab === "calendar" && <MoodCalendar />}
                {activeTab === "statistics" && <MoodStatistics />}
                {activeTab === "settings" && <CustomMoodLabels />}
              </Suspense>
            </main>
          </div>
        </div>
      </div>
    </ErrorBoundary>
  );
}
```

@oluwakemi Oluwadahunsi

Glances of what you should have



Mood Tracker

Today | Calendar | Statistics | Settings

How are you feeling?

Happy | Sad | Calm | Neutral | Excited

Relaxed | Angry | Bright

Add a note (optional)

Log Mood

Activities

Exercise | Good sleep | Work | Socializing | Reading | Meditation

Hobby | Eating | Watching Movies | Gaming | Traveling | Shopping

Listening to Music

Add custom activity

Add Custom Activity

Mood-Based Suggestions for Happy

- Celebrate your good mood
- Share your happiness with others
- Try a new hobby
- Express gratitude
- Set new goals

Recent Entries

October 9, 2024

9:38 AM

Calm

It's a cool day today!

Good sleep | Hobby | Eating | Watching Movies

10:22 AM

Happy

I am glad this project is done!

Work | Hobby | Listening to Music

Mood Tracker

Today | Calendar | Statistics | Settings

Add Custom Mood

Mood Label

Mood Emoji

Choose Emoji

Mood Color (e.g., #FF0000 for red)

Add Custom Mood

Bright

Mood Tracker

Today | Calendar | Statistics | Settings

Add Custom Mood

Mood Label

Mood Emoji

Choose Emoji

Mood Color (e.g., #FF0000 for red)

Add Custom Mood

Bright

Mood Tracker

Today | Calendar | Statistics | Settings

Mood Calendar

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	1	2	3	4	5
6	7	8	9 😊	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2

@oluwakemi Oluwadahunsi

Mood Tracker

Today | Calendar | Statistics | Settings

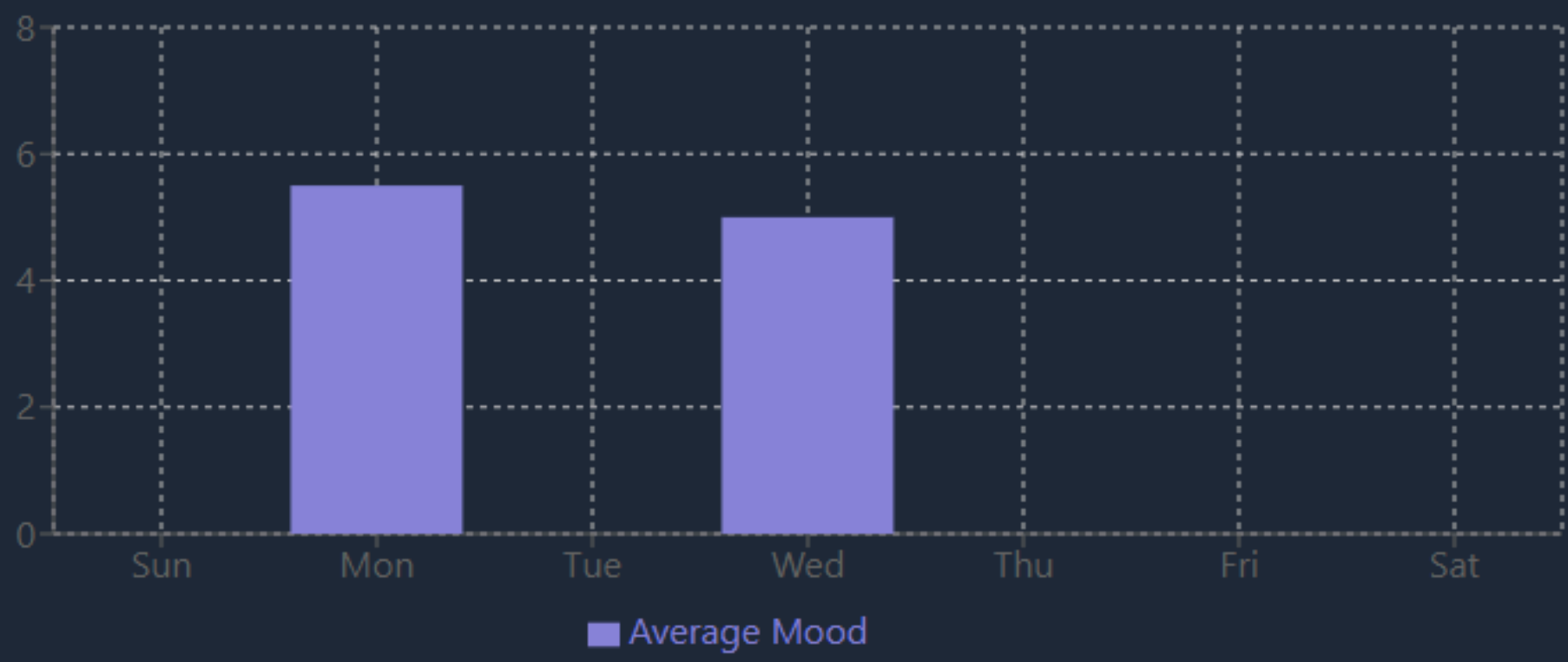
Mood Statistics

Most Frequent Mood

Excited (2 times)

Mood Distribution

Weekly Mood Trend



@oluwakemi Oluwadahunsi

Upload to GitHub

To host our application on free app hosting platforms, we need to create and push it to a Github repository which I have done.

So, upload yours to your Github, after confirming that all is fine and functioning well.

To correct errors and check for updates, visit the project source codes here:

Link:
https://github.com/Kemi-Oluwadahunsi/Mood-Tracker-App-Project

You can deploy your application on any platform of your choice and begin to use.

Congratulations

Did you follow through this project building phase with us, or is it your first time building a react project?

Well, congratulations if you did! 🎉🎉🎉

Last advice: Go through this project thoroughly again and again. Understand properly what each component and function does and how they were used.

Watch out for more updates on this project as I will be taking it a little further in the next few days. The repository will definitely be updated.

Thanks again for staying through, I hope you have a wonderful time building!

Once again, remember to use correct import paths.

@oluwakemi Oluwadahunsi

I hope you found this material useful and helpful.

Remember to:

Like

Save for future reference

&

Share with your network, be helpful to someone 👌

@oluwakemi Oluwadahunsi

Hi There!

Thank you for reading through

Did you enjoy this knowledge?

💼 Follow my LinkedIn page for more work-life balancing and Coding tips.

🌐 LinkedIn: Oluwakemi Oluwadahunsi

kodemaven-portfolio.vercel.app