```tsx
"use client"

import { Button } from "@/components/ui/button"
import { Trash2 } from "lucide-react"
import { Badge } from "@/components/ui/badge"
import { cn } from "@/lib/utils"
import type { Pokemon } from "@/lib/types"

// This interface defines the props for the TeamPreview component.
interface TeamPreviewProps {
  team: Pokemon[]
  onRemove: (pokemon: Pokemon) => void
  onSelect: (pokemon: Pokemon) => void
}

/**
 * This component displays a preview of a Pokémon team, allowing users to see their selected Pokémon,
 * remove Pokémon from the team, and select a Pokémon for further actions.
 * It is designed to be used in a team-building interface where users can manage their Pokémon roster.
 * @param team - An array of Pokémon objects representing the team.
 * @param onRemove - A callback function to handle the removal of a Pokémon from the team.
 * @param onSelect - A callback function to handle the selection of a Pokémon for further actions.
 * @returns
 */
export default function TeamPreview({ team = [], onRemove, onSelect }: TeamPreviewProps) {
  // Ensure the team is an array and fill empty slots if necessary.
  // This ensures that the team is always treated as an array, even if it is undefined or null.
  // It also calculates how many empty slots are needed to fill the team to a total of 6 Pokémon.
  const safeTeam = Array.isArray(team) ? team : [];
  const emptySlots = Array(Math.max(0, 6 - safeTeam.length)).fill(null);

  // This object maps Pokémon types to their corresponding CSS classes for background colors.
  const typeColors: Record<string, string> = {
    normal: "bg-stone-400",
    fire: "bg-orange-500",
    water: "bg-blue-500",
    electric: "bg-yellow-400",
    grass: "bg-green-500",
    ice: "bg-cyan-300",
    fighting: "bg-red-700",
    poison: "bg-purple-600",
    ground: "bg-amber-600",
    flying: "bg-indigo-300",
    psychic: "bg-pink-500",
    bug: "bg-lime-500",
    rock: "bg-yellow-700",
    ghost: "bg-purple-800",
    dragon: "bg-violet-700",
    dark: "bg-stone-700",
    steel: "bg-slate-400",
    fairy: "bg-pink-300",
  }

  return (
    <div className="grid grid-cols-2 gap-3">
      {/* This maps over the Pokémon in the team and renders each one with its sprite, name, types, and a remove button. */}
      {safeTeam.map((pokemon, index) => {
        if (!pokemon) return null;

        const pokemonName = pokemon.name || "Unknown";
        const pokemonTypes = pokemon.types || [];
        const pokemonSprite = pokemon.sprite || "/placeholder.svg?height=48&width=48";
        return (
          <div
            key={`${pokemon.pdx_num}-${index}`}
            className="flex items-center p-2 border rounded-md bg-white dark:bg-slate-800 cursor-pointer hover:bg-slate-50 dark:hover:bg-slate-700 transition-colors"
            onClick={() => onSelect(pokemon)}
          >
            <div className="relative w-12 h-12 mr-2 flex-shrink-0">
              {/* This renders the Pokémon sprite with a fallback to a placeholder image if the sprite is not available. */}
              <img
                src={pokemonSprite || "/placeholder.svg"}
                alt={pokemonName}
                className="w-full h-full object-contain"
                onError={(e) => {
                  const target = e.target as HTMLImageElement
                  target.src = "/placeholder.svg?height=48&width=48"
                }}
              />
            </div>
            <div className="flex-1 min-w-0">
              <h4 className="text-sm font-medium truncate">{pokemonName}</h4>
              <div className="flex gap-1 mt-1 flex-wrap">
                {/* Renders a Badge with the Pokemon Type in an appropriate Color */}
                {pokemonTypes.map((type: string, typeIndex: number) => (
                  <Badge
                    key={`${type}-${typeIndex}`}
                    className={cn("text-white text-[10px] px-1 py-0", typeColors[type?.toLowerCase()] || "bg-gray-500")}
                  >
                    {type}
                  </Badge>
                ))}
              </div>
            </div>
            <Button
              variant="ghost"
              size="icon"
              className="h-7 w-7 ml-1 flex-shrink-0 hover:bg-red-50 dark:hover:bg-red-900/20"
              onClick={(e) => {
                e.stopPropagation();
                onRemove(pokemon);
              }}
            >
              <Trash2 className="h-4 w-4 text-red-500" />
              <span className="sr-only">Remove {pokemonName}</span>
            </Button>
          </div>
        );
      })}
      {/* This maps over the empty slots and renders a placeholder for each empty slot. */}
      {emptySlots.map((_, index) => (
        <div
          key={`empty-${index}`}
          className="flex items-center justify-center p-2 border rounded-md border-dashed h-[72px] bg-slate-50 dark:bg-slate-800/50 transition-colors"
        >
          <span className="text-xs text-muted-foreground">Empty slot</span>
        </div>
      ))}
    </div>
  );
}
```