

```

import React, { useState, useEffect } from "react";
import { base44 } from "@api/base44Client";
import { useQuery, useMutation, useQueryClient } from
"@tanstack/react-query";
import { Button } from "@components/ui/button";
import { Card, CardContent, CardHeader, CardTitle } from
"@components/ui/card";
import { Input } from "@components/ui/input";
import { Label } from "@components/ui/label";
import { Textarea } from "@components/ui/textarea";
import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue
} from "@components/ui/select";
import { Badge } from "@components/ui/badge";
import { format } from "date-fns";
import { Calendar, Video, MapPin, Shield, Clock, User, CheckCircle, X }
from "lucide-react";

const professionalTypes = {
  therapist: { label: "Therapist", icon: "🧑", color: "bg-purple-100
text-purple-800" },
  counselor: { label: "Counselor", icon: "💬", color: "bg-blue-100
text-blue-800" },
  psychiatrist: { label: "Psychiatrist", icon: "🩺", color: "bg-teal-100
text-teal-800" },
  life_coach: { label: "Life Coach", icon: "🎯", color: "bg-green-100
text-green-800" },
  support_group: { label: "Support Group", icon: "👥", color:
"bg-indigo-100 text-indigo-800" }
};

const statusColors = {
  scheduled: "bg-blue-100 text-blue-800",
  completed: "bg-green-100 text-green-800",
  cancelled: "bg-red-100 text-red-800",
  rescheduled: "bg-yellow-100 text-yellow-800"
};

export default function BookSession() {
  const [showForm, setShowForm] = useState(false);
  const [user, setUser] = useState(null);
  const queryClient = useQueryClient();

  const [formData, setFormData] = useState({

```

```

    appointment_date: "",
    professional_type: "therapist",
    professional_name: "",
    session_type: "online",
    notes: "",
    is_anonymous: false,
    duration_minutes: 60
  });

useEffect(() => {
  const fetchUser = async () => {
    try {
      const userData = await base44.auth.me();
      setUser(userData);
    } catch (error) {
      base44.auth.redirectToLogin();
    }
  };
  fetchUser();
}, []);

const { data: appointments = [], isLoading } = useQuery({
  queryKey: ['appointments'],
  queryFn: async () => {
    const appts = await base44.entities.Appointment.filter({
      created_by: user?.email }, '-appointment_date');
    return appts;
  },
  enabled: !!user
});

const createAppointmentMutation = useMutation({
  mutationFn: (data) => base44.entities.Appointment.create(data),
  onSuccess: () => {
    queryClient.invalidateQueries(['appointments']);
    setShowForm(false);
    resetForm();
  }
});

const updateStatusMutation = useMutation({
  mutationFn: ({ id, status }) =>
base44.entities.Appointment.update(id, { status }),

```

```

    onSuccess: () => {
      queryClient.invalidateQueries(['appointments']);
    }
  });

const resetForm = () => {
  setFormData({
    appointment_date: "",
    professional_type: "therapist",
    professional_name: "",
    session_type: "online",
    notes: "",
    is_anonymous: false,
    duration_minutes: 60
  });
};

const handleSubmit = (e) => {
  e.preventDefault();
  createAppointmentMutation.mutate(formData);
};

const upcomingAppointments = appointments.filter(a =>
  a.status === "scheduled" && new Date(a.appointment_date) >= new
Date()
);

const pastAppointments = appointments.filter(a =>
  a.status === "completed" || new Date(a.appointment_date) < new
Date()
);

if (!user) return null;

return (
  <div className="py-12 px-4">
    <div className="max-w-6xl mx-auto space-y-8">
      { /* Header */ }
      <div className="flex flex-col md:flex-row justify-between
items-start md:items-center gap-4">
        <div>
          <h1 className="text-4xl font-bold bg-gradient-to-r
from-purple-600 to-teal-600 bg-clip-text text-transparent">

```

```

        Book a Session
    </h1>
    <p className="text-gray-600 mt-2">Connect with licensed
mental health professionals</p>
</div>
<Button
    onClick={() => setShowForm(!showForm)}
    className="bg-gradient-to-r from-purple-500 to-teal-500
text-white shadow-lg"
>
    {showForm ? "Cancel" : "+ New Appointment"}
</Button>
</div>

{/* Stats */}
<div className="grid md:grid-cols-3 gap-6">
    <Card className="border-none shadow-lg">
        <CardContent className="p-6">
            <div className="flex items-center justify-between">
                <div>
                    <p className="text-sm text-gray-500">Total
Sessions</p>
                    <p className="text-3xl font-bold
text-purple-600">{appointments.length}</p>
                </div>
                <Calendar className="w-10 h-10 text-purple-400" />
            </div>
        </CardContent>
    </Card>

    <Card className="border-none shadow-lg">
        <CardContent className="p-6">
            <div className="flex items-center justify-between">
                <div>
                    <p className="text-sm text-gray-500">Upcoming</p>
                    <p className="text-3xl font-bold
text-blue-600">{upcomingAppointments.length}</p>
                </div>
                <Clock className="w-10 h-10 text-blue-400" />
            </div>
        </CardContent>
    </Card>

```

```

<Card className="border-none shadow-lg">
  <CardContent className="p-6">
    <div className="flex items-center justify-between">
      <div>
        <p className="text-sm text-gray-500">Completed</p>
        <p className="text-3xl font-bold text-green-600">
          {appointments.filter(a => a.status ===
"completed").length}
        </p>
      </div>
      <CheckCircle className="w-10 h-10 text-green-400" />
    </div>
  </CardContent>
</Card>
</div>

```

```

{/* Booking Form */}

```

```

{showForm && (

```

```

  <Card className="border-none shadow-xl">
    <CardHeader>
      <CardTitle>Schedule Your Session</CardTitle>
    </CardHeader>
    <CardContent>
      <form onSubmit={handleSubmit} className="space-y-6">
        <div className="grid md:grid-cols-2 gap-6">
          <div className="space-y-2">
            <Label>Professional Type</Label>
            <Select
              value={formData.professional_type}

```