code attempt

```
import React, { useState, useEffect } from 'react';
import { User, Bell, Menu, X } from 'lucide-react';

// Book + Star logo with rotating star
const BookStarLogo = ({ size = 32, className = "" }) => (
 <svg
  width={size}
  height={size}
  viewBox="0 0 32 32"
  fill="none"
  xmlns="http://www.w3.org/2000/svg"
  className={`${className} drop-shadow-lg`}
 >
  {/* Book - static with crisp white stroke */}
  <path
   d="M7 4V24C7 25.1046 7.89543 26 9 26H25C25.5523 26 26 25.5523 26 25
   stroke="white"
   strokeWidth="2"
   strokeLinejoin="round"
  />
  <path
   d="M7 4C7 5.10457 7.89543 6 9 6H25C25.5523 6 26 6.44772 26 7V25C26 2
   stroke="white"
   strokeWidth="2"
   strokeLinejoin="round"
  />
  <path
   d="M16 6V26"
   stroke="white"
   strokeWidth="1.5"
   strokeLinecap="round"
  />
```

```
    {/* Star with smooth rotation */}
    <path
      d="M16 1L17.2447 4.26184H20.7553L17.7553 6.22652L18.9266 9.48836L16 7
      fill="white"
      stroke="white"
      strokeWidth="0.5"
      strokeLinecap="round"
      strokeLinejoin="round"
      className="animate-spin"
      style={{ animationDuration: '5s', transformOrigin: 'center', animationTimingFu
    />
  </svg>
);

// Custom bell animation that keeps going
const bellPulseKeyframes = `
@keyframes bellPulse {
  0%, 100% { transform: scale(1); }
  50% { transform: scale(1.1); }
}`;

const DocConnectFinalHeader = () => {
  const [language, setLanguage] = useState('fr');
  const [showSidebar, setShowSidebar] = useState(false);
  const [scrolled, setScrolled] = useState(false);
  const unreadQuestionsCount = 1; // Example notification count

  // Add scroll effect
  useEffect(() => {
    const handleScroll = () => {
      const isScrolled = window.scrollY > 10;
      if (isScrolled !== scrolled) {
        setScrolled(isScrolled);
      }
    };
```

```
  window.addEventListener('scroll', handleScroll);
  return () => {
    window.removeEventListener('scroll', handleScroll);
  };
}, [scrolled]);

// Custom CSS for animations and variables
useEffect(() => {
  document.documentElement.style.setProperty('--indigo-500', '#6366f1');
  document.documentElement.style.setProperty('--indigo-600', '#4f46e5');
  document.documentElement.style.setProperty('--cyan-400', '#22d3ee');
  document.documentElement.style.setProperty('--cyan-500', '#06b6d4');

  // Add custom bell animation
  const styleSheet = document.createElement("style");
  styleSheet.type = "text/css";
  styleSheet.innerText = bellPulseKeyframes;
  document.head.appendChild(styleSheet);

  return () => {
    document.head.removeChild(styleSheet);
  };
}, []);

// Custom styles
const headerStyle = {
  backgroundImage: `radial-gradient(circle at top left, rgba(255,255,255,0.1) 0%
};

return (
  <header
    className={`transition-all duration-300 ease-in-out sticky top-0 z-50 ${
      scrolled
        ? 'shadow-lg py-2'
        : 'py-3'
```

```
    }`}
    style={headerStyle}
  >
    {/* Decorative header patterns with softer, blurred circles */}
    <div className="absolute inset-0 overflow-hidden">
      <div className="absolute top-0 left-0 w-full h-full">
        <svg width="100%" height="100%" viewBox="0 0 100 100" preserveAspe
          <circle cx="10" cy="10" r="8" fill="white" style={{ filter: 'blur(2px)' }} />
          <circle cx="40" cy="30" r="6" fill="white" style={{ filter: 'blur(2px)' }} />
          <circle cx="70" cy="70" r="10" fill="white" style={{ filter: 'blur(2px)' }} />
          <circle cx="90" cy="20" r="4" fill="white" style={{ filter: 'blur(2px)' }} />
        </svg>
      </div>
    </div>

    <div className="container mx-auto px-4 relative">
      <div className="flex justify-between items-center">
        <div className="flex items-center">
          <button
            className="lg:hidden mr-3 relative z-10 hover:bg-white/10 p-1 rounded
            onClick={() => setShowSidebar(!showSidebar)}
          >
            {showSidebar ? <X size={24} /> : <Menu size={24} />}
          </button>

          {/* Logo and App Name with enhanced styling */}
          <div className="flex items-center group">
            <div className="relative mr-3">
              <BookStarLogo />
            </div>
            <div>
              <h1 className="text-2xl font-['Poppins','sans-serif'] font-bold tracking
                DocConnect
                <span className="absolute -bottom-1 left-0 w-0 h-0.5 bg-white grou
              </h1>
              <p className="text-xs italic text-white/70 mt-0.5 leading-tight">
```

```
            {language === 'fr' ? 'Connectez-vous à vos professeurs' : 'Connect w
          </p>
        </div>
      </div>
    </div>

    <div className="flex items-center space-x-4">
      {/* Circular Language Toggle buttons */}
      <div className="flex space-x-1">
        <button
          className={`w-8 h-8 text-sm rounded-full flex items-center justify-ce
            language === 'fr'
              ? 'bg-cyan-200 text-indigo-800 transform scale-105 shadow-md for
              : 'bg-white/10 text-white hover:bg-white/20 hover:shadow'
          }`}
          onClick={() => setLanguage('fr')}
          style={language !== 'fr' ? { boxShadow: 'none', hover: { boxShadow: '0
        >
          FR
        </button>
        <button
          className={`w-8 h-8 text-sm rounded-full flex items-center justify-ce
            language === 'en'
              ? 'bg-cyan-200 text-indigo-800 transform scale-105 shadow-md for
              : 'bg-white/10 text-white hover:bg-white/20 hover:shadow'
          }`}
          onClick={() => setLanguage('en')}
          style={language !== 'en' ? { boxShadow: 'none', hover: { boxShadow: '(
        >
          EN
        </button>
      </div>

      {/* Notification Bell with continuous animation when notifications exist */}
      <div className="relative">
        <Bell
```

```
            size={20}
            className="transition-transform hover:scale-110 cursor-pointer"
            style={unreadQuestionsCount > 0 ? {
              animation: 'bellPulse 1.5s ease-in-out infinite',
            } : {}}
          />
          {unreadQuestionsCount > 0 && (
            <span className="absolute -top-1 -right-1 bg-cyan-500 text-white tex
              {unreadQuestionsCount}
            </span>
          )}
        </div>

        {/* User Profile with scale effect on hover */}
        <div className="w-9 h-9 bg-indigo-500/20 rounded-full flex items-cent
          <User size={18} className="text-white" />
        </div>
      </div>
    </div>
  </header>
 );
};

export default DocConnectFinalHeader;
import React, { useState, useEffect } from 'react';
import { Bell, BellRing, MessageSquare, User, Search, Filter, ChevronDown, Chev

const DocConnectStudent = () => {
 const [activeTab, setActiveTab] = useState('all');
 const [searchQuery, setSearchQuery] = useState('');
 const [expandedProf, setExpandedProf] = useState(null);
 const [showQuestionModal, setShowQuestionModal] = useState(false);
 const [selectedProfessor, setSelectedProfessor] = useState(null);
 const [questionText, setQuestionText] = useState('');
 const [showNotification, setShowNotification] = useState(false);
```

```
const [notificationMessage, setNotificationMessage] = useState('');
const [language, setLanguage] = useState('fr');
const [showFilters, setShowFilters] = useState(false);

// Sample professors data
const [professors, setProfessors] = useState([
 {
  id: 1,
  name: 'Dr. Amira Bouzidi',
  department: 'Informatique',
  avatar: null,
  status: 'busy',
  statusUpdatedAt: '10:32',
  location: 'Bâtiment C, Bureau 305',
  officeHours: 'Lun, Mer: 10:00 - 12:00',
  courses: ['Systèmes de Bases de Données', 'Développement Web'],
  notificationEnabled: false
 },
 {
  id: 2,
  name: 'Dr. Khaled Mansour',
  department: 'Génie Électrique',
  avatar: null,
  status: 'away',
  statusUpdatedAt: '09:15',
  location: 'Bâtiment Ingénierie, Bureau 210',
  officeHours: 'Mar, Jeu: 14:00 - 16:00',
  courses: ['Électronique Avancée', 'Systèmes Embarqués'],
  notificationEnabled: false
 },
 {
  id: 3,
  name: 'Dr. Leila Trabelsi',
  department: 'Mathématiques',
  avatar: null,
  status: 'available',
```

```
    statusUpdatedAt: '11:00',
    location: 'Département Math, Bureau 112',
    officeHours: 'Mer, Ven: 09:00 - 11:00',
    courses: ['Analyse II', 'Algèbre Linéaire'],
    notificationEnabled: false
  },
  {
    id: 4,
    name: 'Prof. Omar Ben Salah',
    department: 'Commerce',
    avatar: null,
    status: 'in_meeting',
    statusUpdatedAt: '10:45',
    location: 'Faculté de Gestion, Bureau 405',
    officeHours: 'Lun, Jeu: 13:00 - 15:00',
    courses: ['Principes de Marketing', 'Éthique des Affaires'],
    notificationEnabled: false
  }
]);

// Favorite professors (IDs)
const [favorites, setFavorites] = useState([1, 3]);

// Filter professors based on active tab and search query
const getFilteredProfessors = () => {
  let filtered = professors;

  if (activeTab === 'favorites') {
    filtered = professors.filter(prof => favorites.includes(prof.id));
  } else if (activeTab === 'available') {
    filtered = professors.filter(prof => prof.status === 'available');
  }

  if (searchQuery) {
    filtered = filtered.filter(
      prof => prof.name.toLowerCase().includes(searchQuery.toLowerCase()) ||
```

```
      prof.department.toLowerCase().includes(searchQuery.toLowerCase()) ||
      prof.courses.some(course => course.toLowerCase().includes(searchQuery.t
    );
  }

  return filtered;
};

const toggleFavorite = (id) => {
  if (favorites.includes(id)) {
    setFavorites(favorites.filter(favId => favId !== id));
  } else {
    setFavorites([...favorites, id]);
  }
};

const toggleNotification = (id) => {
  setProfessors(professors.map(prof =>
    prof.id === id
      ? { ...prof, notificationEnabled: !prof.notificationEnabled }
      : prof
  ));

  // Simulate notification being set
  const professor = professors.find(p => p.id === id);
  if (!professor.notificationEnabled) {
    setNotificationMessage(`Vous serez notifié quand ${professor.name} sera di
    setShowNotification(true);
    setTimeout(() => setShowNotification(false), 3000);
  }
};

const handleStatusChange = (id, newStatus) => {
  // This would be triggered by WebSocket in a real implementation
  setProfessors(professors.map(prof =>
    prof.id === id
```

```
      ? { ...prof, status: newStatus, statusUpdatedAt: getCurrentTime() }
      : prof
  ));

  // Check if notification should be triggered
  const professor = professors.find(p => p.id === id);
  if (professor && professor.notificationEnabled && newStatus === 'available') {
    setNotificationMessage(`${professor.name} est maintenant disponible!`);
    setShowNotification(true);
    setTimeout(() => setShowNotification(false), 5000);

    // Reset notification after triggering
    setProfessors(professors.map(prof =>
      prof.id === id
        ? { ...prof, notificationEnabled: false }
        : prof
    ));
  }
};

const getCurrentTime = () => {
  const now = new Date();
  return `${now.getHours()}:${String(now.getMinutes()).padStart(2, '0')}`;
};

const getStatusLabel = (status) => {
  switch(status) {
    case 'available': return language === 'fr' ? 'Disponible' : 'Available';
    case 'busy': return language === 'fr' ? 'Occupé' : 'Busy';
    case 'away': return language === 'fr' ? 'Absent' : 'Away';
    case 'in_meeting': return language === 'fr' ? 'En Réunion' : 'In Meeting';
    default: return status;
  }
};

const getStatusColor = (status) => {
```

```
  switch(status) {
    case 'available': return 'bg-emerald-500';
    case 'busy': return 'bg-rose-500';
    case 'away': return 'bg-amber-500';
    case 'in_meeting': return 'bg-violet-500';
    default: return 'bg-gray-500';
  }
};

const getStatusGradient = (status) => {
  switch(status) {
    case 'available': return 'from-emerald-600 to-teal-500';
    case 'busy': return 'from-rose-600 to-pink-500';
    case 'away': return 'from-amber-600 to-yellow-500';
    case 'in_meeting': return 'from-violet-600 to-purple-500';
    default: return 'from-gray-600 to-gray-500';
  }
};

const openQuestionModal = (prof) => {
  setSelectedProfessor(prof);
  setShowQuestionModal(true);
};

const submitQuestion = () => {
  // In real app, send question to backend
  setNotificationMessage('Votre question a été envoyée');
  setShowNotification(true);
  setTimeout(() => setShowNotification(false), 3000);
  setShowQuestionModal(false);
  setQuestionText('');
};

// Simulate a professor becoming available after 10 seconds
useEffect(() => {
  const timer = setTimeout(() => {
```

```
    handleStatusChange(2, 'available');
  }, 10000);

  return () => clearTimeout(timer);
}, []);

return (
  <div className="flex flex-col h-screen bg-gray-50">
    {/* Navigation bar */}
    <header className="bg-gradient-to-r from-indigo-600 to-violet-600 text-w
      <div className="container mx-auto px-4 py-3">
        <div className="flex justify-between items-center">
          <h1 className="text-2xl font-bold tracking-tight">DocConnect</h1>
          <div className="flex items-center space-x-4">
            <div className="flex bg-white/20 rounded-md overflow-hidden">
              <button
                className={`px-3 py-1 text-sm transition-colors ${language === 'fr'
                onClick={() => setLanguage('fr')}
              >
                FR
              </button>
              <button
                className={`px-3 py-1 text-sm transition-colors ${language === 'en'
                onClick={() => setLanguage('en')}
              >
                EN
              </button>
            </div>
            <div className="w-8 h-8 bg-white/20 rounded-full flex items-center ju
              <User size={18} />
            </div>
          </div>
        </div>
      </div>
    </header>
```

```
      {/* Search and tabs */}
      <div className="bg-white border-b border-gray-200 sticky top-0 z-10 shad
        <div className="container mx-auto px-4 py-3">
          <div className="relative mb-3">
            <div className="absolute inset-y-0 left-0 pl-3 flex items-center pointer-
              <Search size={18} className="text-gray-400" />
            </div>
            <input
              type="text"
              className="block w-full pl-10 pr-10 py-2 border border-gray-300 round
              placeholder={language === 'fr' ? "Rechercher un professeur, départeme
              value={searchQuery}
              onChange={(e) => setSearchQuery(e.target.value)}
            />
            <button
              className="absolute inset-y-0 right-0 pr-3 flex items-center"
              onClick={() => setShowFilters(!showFilters)}
            >
              <Filter size={18} className={`${showFilters ? 'text-indigo-600' : 'text-g
            </button>
          </div>

          {showFilters && (
            <div className="mb-3 p-3 bg-gray-50 rounded-lg border border-gray-2
              <div className="text-sm font-medium mb-2 text-gray-700">
                {language === 'fr' ? 'Filtrer par département' : 'Filter by department'}
              </div>
              <div className="flex flex-wrap gap-2">
                <button className="px-3 py-1 bg-indigo-100 text-indigo-800 rounded
                <button className="px-3 py-1 bg-gray-100 text-gray-800 rounded-fu
                <button className="px-3 py-1 bg-gray-100 text-gray-800 rounded-fu
                <button className="px-3 py-1 bg-gray-100 text-gray-800 rounded-fu
              </div>
            </div>
          )}
```

```
        <div className="flex border-b border-gray-200">
          <button
            className={`px-4 py-2 font-medium text-sm ${activeTab === 'all' ? 'tex
            onClick={() => setActiveTab('all')}
          >
            {language === 'fr' ? 'Tous' : 'All'}
          </button>
          <button
            className={`px-4 py-2 font-medium text-sm ${activeTab === 'favorites
            onClick={() => setActiveTab('favorites')}
          >
            {language === 'fr' ? 'Favoris' : 'Favorites'}
          </button>
          <button
            className={`px-4 py-2 font-medium text-sm ${activeTab === 'available
            onClick={() => setActiveTab('available')}
          >
            {language === 'fr' ? 'Disponibles' : 'Available'}
          </button>
        </div>
      </div>
    </div>

    {/* Professors list */}
    <div className="container mx-auto px-4 py-4 flex-1 overflow-y-auto">
      {getFilteredProfessors().length === 0 ? (
        <div className="flex flex-col items-center justify-center text-center py-10
          <div className="w-16 h-16 bg-gray-200 rounded-full flex items-center j
            <Search size={24} className="text-gray-400" />
          </div>
          <h3 className="text-lg font-medium text-gray-900 mb-1">
            {language === 'fr' ? 'Aucun professeur trouvé' : 'No professors found'}
          </h3>
          <p className="text-gray-500 max-w-md">
            {language === 'fr'
              ? 'Ajustez vos filtres ou essayez une recherche différente'
```

```
            : 'Adjust your filters or try a different search'}
          </p>
        </div>
      ) : (
        <div className="space-y-4">
          {getFilteredProfessors().map((prof) => (
            <div
              key={prof.id}
              className="bg-white rounded-xl border border-gray-200 overflow-hi
            >
              <div className="flex items-center p-4">
                {/* Avatar/Status indicator */}
                <div className="relative mr-3">
                  <div className="w-12 h-12 bg-gradient-to-br from-gray-200 to-gra
                    {prof.name.charAt(0)}
                  </div>
                  <div className={`absolute bottom-0 right-0 w-4 h-4 rounded-full $
                </div>

                {/* Professor info */}
                <div className="flex-1">
                  <div className="flex items-center mb-1">
                    <h3 className="font-medium text-gray-900">{prof.name}</h3>
                    <button
                      className="ml-2 text-gray-400 hover:text-yellow-500"
                      onClick={() => toggleFavorite(prof.id)}
                    >
                      {favorites.includes(prof.id) ? (
                        <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24
                          <path fillRule="evenodd" d="M10.788 3.21c.448-1.077 1.976-1.0
                        </svg>
                      ) : (
                        <svg xmlns="http://www.w3.org/2000/svg" fill="none" viewBox=
                          <path strokeLinecap="round" strokeLinejoin="round" d="M11.48
                        </svg>
                      )}
```

```
                </button>
              </div>
              <div className="flex items-center">
                <p className="text-sm text-gray-500 mr-3">{prof.department}</p
                <div className={`flex items-center px-2 py-1 rounded-full text-xs f
                  <span>{getStatusLabel(prof.status)}</span>
                  <span className="mx-1">•</span>
                  <span>{prof.statusUpdatedAt}</span>
                </div>
              </div>
            </div>

            {/* Action buttons */}
            <div className="flex items-center space-x-2">
              {prof.status !== 'available' && (
                <button
                  className={`p-2 rounded-full ${prof.notificationEnabled ? 'bg-ind
                  onClick={() => toggleNotification(prof.id)}
                  title={language === 'fr' ? "M'avertir quand disponible" : "Notify me
                >
                  {prof.notificationEnabled ? <BellRing size={20} /> : <Bell size={20
                </button>
              )}

              <button
                className="p-2 rounded-full text-gray-400 hover:text-indigo-600
                onClick={() => openQuestionModal(prof)}
                title={language === 'fr' ? "Poser une question" : "Ask a question"}
              >
                <MessageSquare size={20} />
              </button>

              <button
                className="p-2 rounded-full text-gray-400 hover:text-indigo-600
                onClick={() => setExpandedProf(expandedProf === prof.id ? null : p
              >
```

```
                {expandedProf === prof.id ? (
                  <ChevronUp size={20} />
                ) : (
                  <ChevronDown size={20} />
                )}
              </button>
            </div>
          </div>

          {/* Expanded details */}
          {expandedProf === prof.id && (
            <div className="px-4 pb-4 pt-2 border-t border-gray-100">
              <div className="grid grid-cols-1 gap-3">
                <div className="flex items-start text-sm">
                  <MapPin size={16} className="mr-2 text-gray-400 flex-shrink-0
                  <span>{prof.location}</span>
                </div>
                <div className="flex items-start text-sm">
                  <Clock size={16} className="mr-2 text-gray-400 flex-shrink-0 m
                  <span>{prof.officeHours}</span>
                </div>
                <div className="flex items-start text-sm">
                  <BookOpen size={16} className="mr-2 text-gray-400 flex-shrink
                  <div>
                    <p className="font-medium mb-1">
                      {language === 'fr' ? 'Cours enseignés:' : 'Courses taught:'}
                    </p>
                    <ul className="list-disc ml-4 text-gray-600 space-y-1">
                      {prof.courses.map((course, idx) => (
                        <li key={idx}>{course}</li>
                      ))}
                    </ul>
                  </div>
                </div>
              </div>
            </div>
```

```
            )}
          </div>
        ))}
      </div>
    ))}
  </div>

  {/* Question Modal */}
  {showQuestionModal && selectedProfessor && (
    <div className="fixed inset-0 bg-black/50 flex items-end md:items-center
      <div className="bg-white rounded-t-xl md:rounded-xl shadow-xl w-full m
        <div className="flex items-center justify-between p-4 border-b border-
          <div className="flex items-center">
            <button
              className="md:hidden mr-2 text-gray-500"
              onClick={() => setShowQuestionModal(false)}
            >
              <ArrowLeft size={20} />
            </button>
            <h2 className="font-medium text-lg">
              {language === 'fr' ? 'Poser une question' : 'Ask a question'}
            </h2>
          </div>
          <button
            className="text-gray-400 hover:text-gray-500"
            onClick={() => setShowQuestionModal(false)}
          >
            <X size={20} />
          </button>
        </div>

        <div className="p-4">
          <div className="flex items-center mb-4">
            <div className="w-10 h-10 mr-3 bg-gradient-to-br from-gray-200 to-
              {selectedProfessor.name.charAt(0)}
            </div>
```

```
            <div>
              <p className="font-medium">{selectedProfessor.name}</p>
              <p className="text-sm text-gray-500">{selectedProfessor.departme
            </div>
          </div>

          <div className="mb-4">
            <label className="block text-sm font-medium text-gray-700 mb-1">
              {language === 'fr' ? 'Votre question' : 'Your question'}
            </label>
            <textarea
              className="w-full p-3 border border-gray-300 rounded-lg focus:ring
              placeholder={language === 'fr'
                ? "Soyez précis pour obtenir une réponse plus rapide..."
                : "Be specific to get a faster response..."}
              value={questionText}
              onChange={(e) => setQuestionText(e.target.value)}
            ></textarea>
          </div>

          <div className="flex justify-end space-x-3">
            <button
              className="px-4 py-2 border border-gray-300 rounded-lg hover:bg-
              onClick={() => setShowQuestionModal(false)}
            >
              {language === 'fr' ? 'Annuler' : 'Cancel'}
            </button>
            <button
              className="px-4 py-2 bg-indigo-600 text-white rounded-lg hover:bg
              onClick={submitQuestion}
              disabled={!questionText.trim()}
            >
              {language === 'fr' ? 'Envoyer' : 'Send'}
            </button>
          </div>
        </div>
```

```
        </div>
      </div>
    )}

    {/* Toast notification */}
    {showNotification && (
      <div className="fixed bottom-4 left-1/2 transform -translate-x-1/2 bg-gray
        <div className="mr-2 text-indigo-300">
          <BellRing size={18} />
        </div>
        <p>{notificationMessage}</p>
      </div>
    )}
  </div>
 );
};

export default DocConnectStudent;import React, { useState } from 'react';
import { User, Clock, Calendar, MessageSquare, Bell, Settings, LogOut, Menu, X,

const DocConnectSimpleProf = () => {
 const [currentStatus, setCurrentStatus] = useState('busy');
 const [language, setLanguage] = useState('fr');
 const [showSidebar, setShowSidebar] = useState(false);
 const [activeTab, setActiveTab] = useState('status');
 const [waitingStudents, setWaitingStudents] = useState(2);

 // Questions
 const unreadQuestionsCount = 1;

 const getStatusColor = (status) => {
  switch(status) {
   case 'available': return 'bg-emerald-500';
   case 'busy': return 'bg-rose-500';
   case 'away': return 'bg-amber-500';
   case 'in_meeting': return 'bg-violet-500';
```

```
    default: return 'bg-gray-500';
  }
};

const getStatusLabel = (status) => {
  switch(status) {
    case 'available': return language === 'fr' ? 'Disponible' : 'Available';
    case 'busy': return language === 'fr' ? 'Occupé' : 'Busy';
    case 'away': return language === 'fr' ? 'Absent' : 'Away';
    case 'in_meeting': return language === 'fr' ? 'En Réunion' : 'In Meeting';
    default: return status;
  }
};

const updateStatus = (newStatus) => {
  if (newStatus === currentStatus) return;
  setCurrentStatus(newStatus);
  // If changing to available and students are waiting for notification
  if (newStatus === 'available' && waitingStudents > 0) {
    // Reset waiting students
    setWaitingStudents(0);
  }
};

return (
  <div className="flex flex-col h-screen bg-gray-50">
    {/* Top Navigation */}
    <header className="bg-gradient-to-r from-violet-600 to-indigo-600 text-w
      <div className="container mx-auto px-4 py-3">
        <div className="flex justify-between items-center">
          <div className="flex items-center">
            <button
              className="lg:hidden mr-3"
              onClick={() => setShowSidebar(!showSidebar)}
            >
              {showSidebar ? <X size={24} /> : <Menu size={24} />}
```

```
          </button>
          <h1 className="text-2xl font-bold tracking-tight">DocConnect</h1>
        </div>

        <div className="flex items-center space-x-4">
          <div className="flex bg-white/20 rounded-md overflow-hidden">
            <button
              className={`px-3 py-1 text-sm transition-colors ${language === 'fr'
              onClick={() => setLanguage('fr')}
            >
              FR
            </button>
            <button
              className={`px-3 py-1 text-sm transition-colors ${language === 'en'
              onClick={() => setLanguage('en')}
            >
              EN
            </button>
          </div>

          <div className="relative">
            <Bell size={20} />
            {unreadQuestionsCount > 0 && (
              <span className="absolute -top-1 -right-1 bg-rose-500 text-white te
                {unreadQuestionsCount}
              </span>
            )}
          </div>

          <div className="w-8 h-8 bg-white/20 rounded-full flex items-center ju
            <User size={18} />
          </div>
        </div>
      </div>
    </header>
```

```
<div className="flex flex-1 overflow-hidden">
 {/* Sidebar */}
 <aside className={`${showSidebar ? 'block' : 'hidden'} lg:block w-64 bg-
  <div className="flex flex-col h-full">
   <div className="flex flex-col items-center p-6 border-b border-gray-10
    <div className="w-20 h-20 bg-gradient-to-br from-indigo-100 to-viole
     <span className="text-2xl font-bold text-violet-700">A</span>
    </div>
    <h2 className="text-lg font-medium">Dr. Amira Bouzidi</h2>
    <p className="text-sm text-gray-500">Professeur en Informatique</p
   </div>

   <nav className="flex-1 p-4 space-y-1">
    <button
     onClick={() => setActiveTab('status')}
     className={`flex items-center w-full px-3 py-2 text-left rounded-lg ${
    >
     <Clock size={18} className="mr-3" />
     <span>{language === 'fr' ? 'Statut' : 'Status'}</span>
    </button>

    <button
     onClick={() => setActiveTab('questions')}
     className={`flex items-center w-full px-3 py-2 text-left rounded-lg ${
    >
     <MessageSquare size={18} className="mr-3" />
     <span>{language === 'fr' ? 'Questions' : 'Questions'}</span>
     {unreadQuestionsCount > 0 && (
      <span className="ml-auto bg-rose-500 text-white text-xs rounded-
       {unreadQuestionsCount}
      </span>
     )}
    </button>

    <button
```

```
            onClick={() => setActiveTab('appointments')}
            className={`flex items-center w-full px-3 py-2 text-left rounded-lg ${
          >
            <Calendar size={18} className="mr-3" />
            <span>{language === 'fr' ? 'Rendez-vous' : 'Appointments'}</span>
          </button>

          <button
            onClick={() => setActiveTab('settings')}
            className={`flex items-center w-full px-3 py-2 text-left rounded-lg ${
          >
            <Settings size={18} className="mr-3" />
            <span>{language === 'fr' ? 'Paramètres' : 'Settings'}</span>
          </button>
        </nav>

        <div className="p-4 border-t border-gray-100">
          <button className="flex items-center w-full px-3 py-2 text-left rounded
            <LogOut size={18} className="mr-3" />
            <span>{language === 'fr' ? 'Déconnexion' : 'Sign Out'}</span>
          </button>
        </div>
      </div>
    </aside>

    {/* Main Content */}
    <main className="flex-1 overflow-y-auto">
      <div className="container mx-auto px-4 py-6">
        {/* Status Section */}
        {activeTab === 'status' && (
          <div>
            <h2 className="text-xl font-bold mb-4">
              {language === 'fr' ? 'Statut actuel' : 'Current Status'}
            </h2>

            <div className="bg-white rounded-xl shadow-sm border border-gray
```

```
            <div className="p-5 flex items-center">
              <div className={`w-4 h-4 rounded-full ${getStatusColor(currentSta
              <div className="text-lg font-medium">{getStatusLabel(currentStat
            </div>

            <div className="grid grid-cols-2 gap-3 p-5 border-t border-gray-10
              <button
                className={`p-4 rounded-lg flex flex-col items-center justify-cent
                onClick={() => updateStatus('available')}
              >
                <span className="font-medium">{language === 'fr' ? 'Disponible
              </button>

              <button
                className={`p-4 rounded-lg flex flex-col items-center justify-cent
                onClick={() => updateStatus('busy')}
              >
                <span className="font-medium">{language === 'fr' ? 'Occupé' : '
              </button>

              <button
                className={`p-4 rounded-lg flex flex-col items-center justify-cent
                onClick={() => updateStatus('away')}
              >
                <span className="font-medium">{language === 'fr' ? 'Absent' : 'A
              </button>

              <button
                className={`p-4 rounded-lg flex flex-col items-center justify-cent
                onClick={() => updateStatus('in_meeting')}
              >
                <span className="font-medium">{language === 'fr' ? 'En Réunion
              </button>
            </div>

            {/* Students waiting for notification */}
```

```
          {waitingStudents > 0 && currentStatus !== 'available' && (
            <div className="px-5 py-3 bg-indigo-50 border-t border-indigo-10
              <Bell size={16} className="mr-2 text-indigo-500" />
              <span className="text-indigo-700">
                {language === 'fr'
                  ? `${waitingStudents} étudiants attendent que vous soyez dispon
                  : `${waitingStudents} students waiting for you to become availabl
              </span>
            </div>
          )}
        </div>
      </div>
    )}

    {/* Questions Section */}
    {activeTab === 'questions' && (
      <div>
        <h2 className="text-xl font-bold mb-4">
          {language === 'fr' ? 'Questions des étudiants' : 'Student Questions'}
        </h2>

        <div className="bg-white rounded-xl shadow-sm border border-gray
          <div className="p-4 border-b border-gray-100">
            <div className="flex justify-between items-start">
              <div>
                <h3 className="font-medium flex items-center">
                  Sami Gharbi
                  <span className="ml-2 bg-indigo-100 text-indigo-800 text-xs p
                    {language === 'fr' ? 'Nouveau' : 'New'}
                  </span>
                </h3>
                <p className="text-sm text-gray-500">Systèmes de Bases de D
              </div>
            </div>

            <p className="text-sm mt-2">
```

```
            Bonjour Dr. Amira, je n'arrive pas à comprendre le concept de norm
          </p>
        </div>

        <div className="p-3 flex justify-end bg-gray-50">
          <button className="px-4 py-2 bg-indigo-600 text-white rounded-lg
            {language === 'fr' ? 'Répondre' : 'Reply'}
          </button>
        </div>
      </div>
    </div>
  )}

  {/* Appointments Section */}
  {activeTab === 'appointments' && (
    <div>
      <h2 className="text-xl font-bold mb-4">
        {language === 'fr' ? 'Rendez-vous à venir' : 'Upcoming Appointments
      </h2>

      <div className="space-y-4">
        <div className="bg-white rounded-xl shadow-sm border border-gra
          <div className="flex justify-between items-start">
            <h3 className="font-medium">Sara Jouini</h3>
            <div className="text-indigo-600 font-medium">
              14:30 - 15:00
            </div>
          </div>
          <p className="text-sm text-gray-500 mt-1">Aujourd'hui</p>
          <div className="mt-3">
            <p className="text-sm"><span className="font-medium">Cours
            <p className="text-sm"><span className="font-medium">Sujet
          </div>
        </div>

        <div className="bg-white rounded-xl shadow-sm border border-gra
```

```
              <div className="flex justify-between items-start">
                <h3 className="font-medium">Ahmed Triki</h3>
                <div className="text-indigo-600 font-medium">
                  10:00 - 10:30
                </div>
              </div>
              <p className="text-sm text-gray-500 mt-1">Demain</p>
              <div className="mt-3">
                <p className="text-sm"><span className="font-medium">Cours
                <p className="text-sm"><span className="font-medium">Sujet
              </div>
            </div>
          </div>
        </div>
      )}

      {/* Settings Section */}
      {activeTab === 'settings' && (
        <div>
          <h2 className="text-xl font-bold mb-4">
            {language === 'fr' ? 'Paramètres' : 'Settings'}
          </h2>

          <div className="bg-white rounded-xl shadow-sm border border-gray
            <h3 className="font-medium mb-4">{language === 'fr' ? 'Informati

            <div className="space-y-4">
              <div>
                <label className="block text-sm font-medium text-gray-700 mb-1
                  {language === 'fr' ? 'Nom complet' : 'Full Name'}
                </label>
                <input
                  type="text"
                  className="w-full p-2 border border-gray-300 rounded-md"
                  value="Dr. Amira Bouzidi"
                />
```

```
              </div>

              <div>
                <label className="block text-sm font-medium text-gray-700 mb-1
                  {language === 'fr' ? 'Email' : 'Email'}
                </label>
                <input
                  type="email"
                  className="w-full p-2 border border-gray-300 rounded-md"
                  value="amira.bouzidi@univ-tunis.tn"
                />
              </div>

              <div>
                <label className="block text-sm font-medium text-gray-700 mb-1
                  {language === 'fr' ? 'Département' : 'Department'}
                </label>
                <input
                  type="text"
                  className="w-full p-2 border border-gray-300 rounded-md"
                  value="Informatique"
                />
              </div>

              <div className="flex justify-end">
                <button className="px-4 py-2 bg-indigo-600 text-white rounded-
                  {language === 'fr' ? 'Enregistrer' : 'Save'}
                </button>
              </div>
            </div>
          </div>
        </div>
      )}
    </div>
  </main>
</div>
```

```
    </div>
  );
};

export default DocConnectSimpleProf;
```