import React, { useState, useEffect } from "react";

import { DataGrid, GridColDef } from "@mui/x-data-grid";

import { supabase } from "./supabaseClient";

import {

Snackbar,

Alert,

Box,

Typography,

Button,

Dialog,

DialogActions,

DialogContent,

DialogTitle,

Card,

CardContent,

} from "@mui/material";

import CharacterModal\_duocsi from "./CharacterModal\_duocsi";

interface ToaThuoc {

khambenh\_id: string;

ngaytoa: string;

benhnhan\_id: string;

ten\_benhnhan: string;

chan\_doan: string;

thuoc\_id: string;

ten\_thuoc: string;

don\_vi: string;

so\_lan\_dung: number;

so\_luong\_moi\_lan: number;

tong\_so\_luong: number;

ten\_bac\_si: string;

trang\_thai: string;

}

const PharmacistToaManager: React.FC = () => {

const [toaThuocList, setToaThuocList] = useState<ToaThuoc[]>([]);

const [selectedPatient, setSelectedPatient] = useState<ToaThuoc | null>(null);

const [openModal, setOpenModal] = useState(false);

const [openSnackbar, setOpenSnackbar] = useState(false);

const [hasPendingToa, setHasPendingToa] = useState(false);

useEffect(() => {

const fetchToaThuoc = async () => {

const { data, error } = await supabase.from("v\_toaduocsi").select();

if (error) {

console.error("Lỗi lấy toa thuốc:", error);

} else if (data) {

setToaThuocList(data as ToaThuoc[]);

const hasPending = data.some(

(toa: ToaThuoc) => toa.trang\_thai === "Chờ"

);

setHasPendingToa(hasPending);

if (data.length > 0) setOpenSnackbar(true);

}

};

fetchToaThuoc();

}, []);

const handleCloseSnackbar = () => {

setOpenSnackbar(false);

};

const handleOpenModal = (patient: string) => {

const patientToa = toaThuocList.find((toa) => toa.ten\_benhnhan === patient);

setSelectedPatient(patientToa || null);

setOpenModal(true);

};

const handleCloseModal = () => {

setOpenModal(false);

};

const markAsDispensed = async (khambenh\_id: string) => {

const { error } = await supabase

.from("toathuoc")

.update({ trang\_thai: "Đã cấp toa" })

.eq("khambenh\_id", khambenh\_id);

if (error) {

console.error("Lỗi cập nhật trạng thái:", error);

} else {

const { data, error: fetchError } = await supabase

.from("v\_toaduocsi")

.select();

if (fetchError) {

console.error("Lỗi lấy lại dữ liệu:", fetchError);

} else {

setToaThuocList(data as ToaThuoc[]);

}

setOpenModal(false);

}

};

const columns: GridColDef[] = [

{ field: "ten\_benhnhan", headerName: "Tên bệnh nhân", width: 200 },

{ field: "trang\_thai", headerName: "Trạng thái", width: 150 },

];

const uniquePatients = Array.from(

new Set(toaThuocList.map((item) => item.ten\_benhnhan))

);

return (

<Box px={0} py={0.5}>

{hasPendingToa && <CharacterModal\_duocsi />}

<Snackbar

open={openSnackbar}

autoHideDuration={6000}

onClose={handleCloseSnackbar}

>

<Alert

onClose={handleCloseSnackbar}

severity="info"

sx={{ width: "100%" }}

>

Có toa thuốc mới cần cấp!

</Alert>

</Snackbar>

{/\* Header: sát lề trái, giảm khoảng cách top, padding và font-size \*/}

<Box

sx={{

backgroundColor: "#e0b3f0",

p: "10px", //

borderRadius: "4px",

width: "100%",

ml: 0, // sát lề trái

mt: 0.8, // cách bảng trên ~4px

mb: 0.5, // cách DataGrid ~4px (nếu muốn)

}}

>

<Typography

variant="h2"

sx={{

fontSize: "1.5rem", // giảm 35% so với 1.5rem mặc định

m: 0.5, // xóa margin

lineHeight: 1.2,

}}

>

Danh sách bệnh nhân chờ cấp thuốc

</Typography>

</Box>

<DataGrid

rows={uniquePatients.map((name, index) => ({

id: index,

ten\_benhnhan: name,

trang\_thai:

toaThuocList.find((toa) => toa.ten\_benhnhan === name)?.trang\_thai ||

"Chờ",

}))}

columns={columns}

autoHeight

onRowClick={(params) => handleOpenModal(params.row.ten\_benhnhan)}

sx={{

ml: 0, // đảm bảo DataGrid cũng sát trái

mt: 0.5, // không thêm khoảng cách trên

mb: 2, // khoảng cách dưới nếu cần dialog hoặc phần khác

}}

/>

{selectedPatient && (

<Dialog open={openModal} onClose={handleCloseModal}>

<DialogTitle>

Toa thuốc của: {selectedPatient.ten\_benhnhan}

</DialogTitle>

<DialogContent>

<Card sx={{ marginBottom: 2 }}>

<CardContent>

<Typography variant="h6">

Chẩn đoán: {selectedPatient.chan\_doan}

</Typography>

<Typography>Ngày khám: {selectedPatient.ngaytoa}</Typography>

<Typography>Bác sĩ: {selectedPatient.ten\_bac\_si}</Typography>

<Typography>

Trạng thái: {selectedPatient.trang\_thai}

</Typography>

</CardContent>

</Card>

{toaThuocList

.filter(

(toa) => toa.ten\_benhnhan === selectedPatient.ten\_benhnhan

)

.map((toa) => (

<Card key={toa.khambenh\_id} sx={{ marginBottom: 2 }}>

<CardContent>

<Typography variant="body1">

{toa.ten\_thuoc} - {toa.don\_vi} - {toa.so\_lan\_dung} lần -{" "}

{toa.so\_luong\_moi\_lan} mỗi lần - Tổng: {toa.tong\_so\_luong}

</Typography>

</CardContent>

</Card>

))}

</DialogContent>

<DialogActions>

<Button

variant="contained"

color="primary"

onClick={() => markAsDispensed(selectedPatient.khambenh\_id)}

>

Đã phát thuốc

</Button>

<Button onClick={handleCloseModal} color="primary">

Đóng

</Button>

</DialogActions>

</Dialog>

)}

</Box>

);

};

export default PharmacistToaManager;