mport \* as z from "zod";

import axios from "axios";

// import MuxPlayer from "@mux/mux-player-react";

import { Pencil, PlusCircle, Video } from "lucide-react";

import { useState } from "react";

import toast from "react-hot-toast";

import { useRouter } from "next/navigation";

import { Chapter, MuxData } from "@prisma/client";

import Image from "next/image";

import { zodResolver } from "@hookform/resolvers/zod";

import { useForm } from "react-hook-form";

import { Input } from "@/components/ui/input";

import { Button } from "@/components/ui/button";

import {

Form,

FormControl,

FormField,

FormItem,

FormMessage,

} from "@/components/ui/form";

interface ChapterVideoFormProps {

initialData: Chapter & { muxData?: MuxData | null };

courseId: string;

chapterId: string;

};

const formSchema = z.object({

videoUrl: z.string().min(1),

});

// const formSchema = z.object({

// title: z.string().min(1),

// });

export const ChapterVideoForm = ({

initialData,

courseId,

chapterId,

}: ChapterVideoFormProps) => {

const [isEditing, setIsEditing] = useState(false);

const toggleEdit = () => setIsEditing((current) => !current);

const router = useRouter();

const form = useForm<z.infer<typeof formSchema>>({

resolver: zodResolver(formSchema),

defaultValues: initialData,

});

const { isSubmitting, isValid } = form.formState;

const onSubmit = async (values: z.infer<typeof formSchema>) => {

try {

await axios.patch(/api/courses/${courseId}/chapters/${chapterId}, values);

toast.success("Chapter updated");

toggleEdit();

router.refresh();

} catch {

toast.error("Something went wrong");

}

}

console.log(initialData)

return (

<div className="mt-6 border bg-slate-100 rounded-md p-4">

<div className="font-medium flex items-center justify-between">

Chapter video

<Button onClick={toggleEdit} variant="ghost">

{isEditing && (

<>Cancel</>

)}

{!isEditing && !initialData.videoUrl && (

<>

<PlusCircle className="h-4 w-4 mr-2" />

Add a video

</>

)}

{!isEditing && initialData.videoUrl && (

<>

<Pencil className="h-4 w-4 mr-2" />

Edit video

</>

)}

</Button>

</div>

{!isEditing && (

!initialData.videoUrl ? (

<div className="flex items-center justify-center h-60 bg-slate-200 rounded-md">

<Video className="h-10 w-10 text-slate-500" />

</div>

) : (

<div className="relative aspect-video mt-2">

<iframe

className="video"

width="500"

height="300"

src= {initialData.videoUrl}

title="YouTube video player"

allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share"

allowFullScreen

></iframe>

</div>

)

)}

{isEditing && (

<div>

<Form {...form}>

<form

onSubmit={form.handleSubmit(onSubmit)}

className="space-y-4 mt-4"

>

<FormField

control={form.control}

name="videoUrl"

render={({ field }) => (

<FormItem>

<FormControl>

<Input

disabled={isSubmitting}

placeholder="e.g. 'Introduction to the course'"

{...field}

/>

</FormControl>

<FormMessage />

</FormItem>

)}

/>

<div className="flex items-center gap-x-2">

<Button

disabled={!isValid || isSubmitting}

type="submit"

>

Save

</Button>

</div>

</form>

</Form>

<div className="text-xs text-muted-foreground mt-4">

Upload this chapter&apos;s video

</div>

</div>

)}

{initialData.videoUrl && !isEditing && (

<div className="text-xs text-muted-foreground mt-2">

Videos can take a few minutes to process. Refresh the page if video does not appear.

</div>

)}

</div>

)

}