The task consists of two parts.

First part:

Sign-in endpoint:
POST
URL: /signin/
Unauthorized
Body:{
email: string,
password: string
}
Response:{
success: bool
data :{
token: string
refresh_token: string,
}
}

Category endpoints:
The admin can add/edit/delete/view the categories (please use django admin)
Categories model should contains id as uid, name as string , type (type_1&type_2) as enum, created_at as datetime, updated_at as datetime

GET
URL: /post/categories?name={}type={}&limit={}&offset={}
description: this endpoint will return the list of categories and the user can filter the posts by type and name and must have a paging
Unauthorized
Response:{
success: bool
data :[{
Id: uuid
name: string,
type: string,
created_at: datetime,
updated_at : datetime
}]
}
}

Posts endpoints:
POST Create

URL : /post/
body : {
	title: string,
	description: string,
	created_at: datetime,
	updated_at : datetime,
	category_id: uid // foreign key of category
}
Response:{
	success: bool
	data{
		title: string,
		description: string,
		created_at: datetime,
		updated_at : datetime
		category: {
			Id: uuid
			name: string,
			type: string
		}
	}
}

PUT Update
URL : /post/{slug}
body : {
	title: string,
	description: string,
	created_at: datetime,
	updated_at : datetime,
	category_id: uid // foreign key of category
}
Response:{
	success: bool
	data{
		title: string,
		description: string,
		created_at: datetime,
		updated_at : datetime
		Category: {
			Id: uuid,
			name: string,
			type: string
		}

```
        }
}
```

GET
URL: /post/get?search={}category={}&limit={}&offset={}
description: this endpoint will return the list of posts and the user can filter the posts by category and search (in post title and description) and must have a paging
Unauthorized

```
Response:{
        success: bool
        data :[{
                title: string,
                description: string,
                created_at: datetime,
                updated_at : datetime
                Category: {
                Id: uuid
                name: string,
                type: string
                }]
        }
}
```

Second Part:
Create an endpoint to take a link from Facebook or Twitter and convert it to audio, then upload it to any bucket or cloud and return the audio link.

```
POST /convert_to_audio/
Body{
        video_url: string
}
Response{
        success: bool,
        data{
                audio_url: string
        }
}
```

Please use Swagger as it's an interactive API documentation.
Please use JWT to authenticate the endpoints.
You should send us the login credentials (any fixed email and password), if there is no signup endpoint.
Note! You should authenticate the endpoints that don't have the "Unauthorized" badge.