PS Proyecto intermedio

Tecnologías a utilizar | **Django**, **React**.

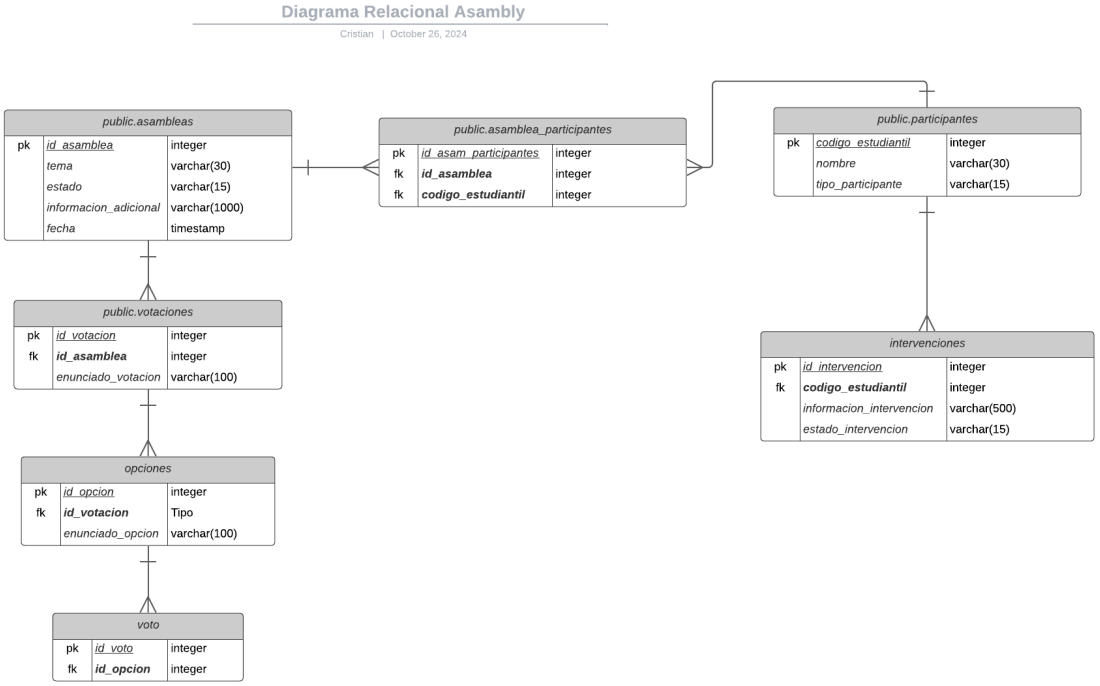
¿Qué debo hacer?

1. Crear un CRUD, para la gestión de las asambleas universitarias. Una asamblea tiene una serie de características como las siguientes:
    a. **Participantes**.
    b. **Decisiones tomadas** durante la asamblea.
    c. **Fecha** de realización de la asamblea.
    d. **Tema** a tratar durante la asamblea.
    e. **Estado:**
        - Programada
        - En proceso
        - Finalizada
    f. Se debe poder registrar a asistentes a la asamblea de manera anónima o con sus datos. De acuerdo a cada participante se podrán registrar sus **intervenciones** (Palabras, mociones). Por último, las **mociones** del asistente deberán tener un estado, Aprobadas, Rechazadas o En Espera.

    Se podrán crear cuantas asambleas se necesiten, se podrán registrar **participantes** y crear votaciones para definir el estado de sus **mociones**.

    1. Se podrán consultar la información de las **asambleas existentes,** juntos con toda su información relacionada.
    2. De igual forma, se podrá actualizar o **eliminar** la información relacionada a una asamblea, el estado en el cual está la asamblea, sus participantes y votaciones.

Diagrama relacional de Asambly.



Vistas que debo definir.

1. **Vista para la creación de una asamblea:** Esta vista tiene como objetivo recibir a partir de un método **POST** los datos principales de la asamblea, fecha, tema y estado. De igual forma, tendrá la capacidad de crear cuantos participantes se requieran.
    a. Deberá tener un **serializer** asociado en el cual se podrá recibir los datos de la asamblea y de los participantes, para asociarlos **uno a uno.** Por último el serializer, tendrá la capacidad de añadir las mociones del participante y el estado de la mismas, iterando uno a uno y dando su *primary_key* de código_estudiantil.
        i. ⚠ **Importante:** Se utilizará el método get_or_create para determinar si el participante ya está creado en la tabla *participantes,* ya que es una relación muchos a muchos entre *asambleas y participantes.* En la tabla *asamblea_participantes* se relacionarán si un participante tiene varias asambleas a las que ha asistido sin la necesidad de crear nuevamente el participante. ¿Cómo se realizará? Se hará uso del código estudiantil ya que es un **identificador único.**
    b. Por último, el mismo serializer podrá agregar las votaciones, utilizando la tabla *votaciones,* relacionando a esta unas *opciones* y por último un *voto.*

2. **Vista para buscar las asambleas existentes:** Esta vista tiene como objetivo recibir un método **GET**, y retornar todas las asambleas existentes.
    a. 🎨 En el front, se utilizarán unas cards, para mostrar el total de las asambleas y una búsqueda para encontrar alguna a partir de su nombre.
3. **Vista para modificar(UPDATE) una asamblea:** Esta vista deberá recibir un *id_asamblea* almacenado en el front y la información completa de la asamblea. El *id* será recibido a partir de la url. El back tendrá las características suficientes para almacenarlo.
    a. ⚠ **Importante:** En el serializer se definirá un función para la actualización de los datos. Está función deberá recibir la instancia u objeto de la base de datos que coincide con el Id, y toda la data que ha sido pasada. Adentro de esta función se involucrará la tabla *asamblea_participantes,* para eliminar los participantes y redefinirlos, creandolos o actualizandolos de la tabla *participantes.*
        i. Toda la información relacionada a la asamblea será actualizada de una manera similar. Únicamente habrá una **restricción** con la tabla de votos, que solo será actualizada si cambian los votos o si se elimina una votación u opción.
4. **Vista para eliminar una asamblea(O sus datos):** Para eliminar una asamblea se deberá definir únicamente la vista, esta vista recibirá el id de la asamblea en la url, la vista buscará la asamblea, y por el ***constrict*** **CASCADE,** todos los elementos relacionados se eliminarán.
    a. En el caso de eliminar participantes o votaciones, se definirá una vista **alterna**, que recibirá, en el caso de los participantes, los *código_estudiantil* de los estudiantes que se eliminarán de la tabla *asamblea_participantes.*
    b. ⚠ **Importante:** Para ambos cosas se utilizará el método **DELETE.**