Segunda entrega:

Modelo de datos e implementación de la Base de Datos

Julio Díaz López

Objetivo: Detallar el modelo de datos para dar funcionalidad a la aplicación Surf Skate Spot, y argumentar la implementación del tipo elegido de base de datos, características y ventajas.

1. Modelo de datos:

- Detalle de Entidades y atributos:

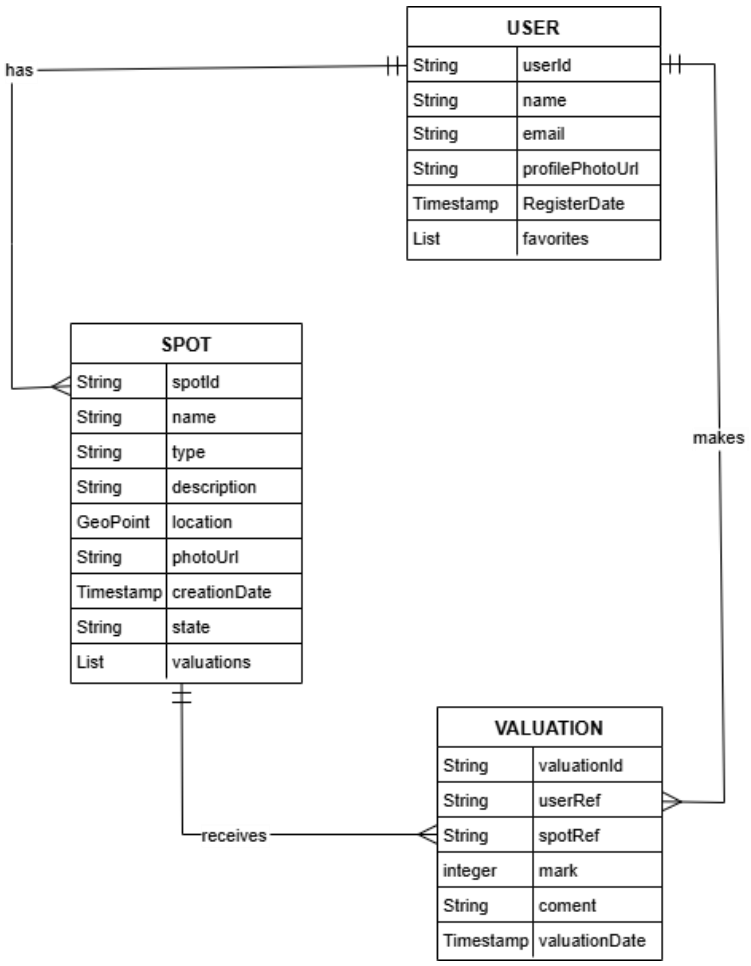
  - User:
    - userId: String – UID Firebase Authentication
    - name: String
    - email: String
    - profilePhotoUrl: String – Opcional
    - favorites: List<String> (spotId)
    - registerDate: Timestamp

  - Spot:
    - spotId: String (ID)
    - userRef: DocumentReference (vinculo al usuario que lo creó)
    - name: String
    - type: Strign
    - description: String
    - location: GeoPoint
    - photoUrl
    - creationDate
    - state: String (Enum: activo en buen estado, activo en mal estado, inactivo)
    - valuations: List<String> (valuationsId, con valoraciones relacionadas)

- Valuation:
    - valuationId: String (ID)
    - userRef: DocumentReference (referencia al creador)
    - spotRef: DocumentReference (referencia al spot)
    - mark: int {1,5}
    - coment: String
    - valuationDate: Timestamp

- Relaciones:
    - Un User puede crear muchos Spot (1:n)
    - Un Spot puede tener varias Valuation en relación (1:n)
    - Un User puede emitir una única valoración al año por spot.

- Diagrama E-R



2. Implementación en Firebase Firestore
   - Tecnología seleccionada:
     - Firebase Firestore (Base de datos NoSQL en tiempo real)
     - Tipos nativos: GeoPoint y Timestamp
     - Integración nativa con Android y Kotlin y escalabilidad mas sencilla.
   - Ejemplo de formato JSON
     - Para reflejar la forma en la que Firestore guarda los objetos JSON, pego un ejemplo de como será la estructura interna de la base de datos.

```json
[
  {
    "spotId": "SP004",
    "userRef": "users/usuarioXYZ",
    "name": "Pumptrack Loredo",
    "type": "Pumptrack",
    "description": "El mas largo de la zona, rapido y con giros fuertes.",
    "location": {
      "_latitude": 43.4792,
      "_longitude": -3.7749
    },
    "photoUrl": "https://ejemplo.com/imagenes/loredo1.jpg",
    "creationDate": "2024-07-15T10:30:00Z",
    "state": "activo en buen estado",
    "valuations": [
      "VAL001",
      "VAL005",
      "VAL008"
    ]
  }
]
```

   - Reglas de seguridad y diseño de índices
     - Documentos de usuario. Solo un usuario autenticado puede ver cualquier spot completo. Solo el usuario creador puede borrar el spot o actualizarlo. Así cada cuenta es autónoma.
     - Favoritos. Subcoleccion que solo puede actualizar el propio usuario de la colección.
     - Spots: Cualquier usuario autenticado puede ver cualquier spot disponible completo. Solo un usuario autenticado puede crear un spot, y solo el creador puede borrarlo o actualizarlo.
     - Valoración. Solo los usuarios autenticados puede hacer valoraciones, con la restricción de una por spot y año. Solo el administrador puede borrar valoraciones.

- Para optimizar rendimiento de consultas, preparo índices compuestos para facilitar las ordenaciones habituales.
    - Indice de spots. Entre type y state, para filtrar rápido por categoría y representar en el mapa.
    - Indice de valuations. Entre spotRef y mark, para generación de rankings sin dale vueltas a toda la colección.