59. Entity - Updoot.ts

#typeorm #graphql #entity #manytomany #backend

Implement Updoot entity

- Now we implement the Updoot entity which will store the upvotes of a post and define a many-to-many relationship between User and Post entities
- Many users can upvote a post AND a user can upvote many posts, therefore many-to-many
- userID and postId are the primary keys, so each updoot is unique based on these two keys
- Note that we're not exposing any of the fields to graphql since client does not need to see this entity and will *only* need the points field in the Post entity

/entities/Updoot.ts

```
import { Field, ObjectType } from "type-graphql";
import { BaseEntity, Column, Entity, ManyToOne, PrimaryColumn } from "typeorm";
import { Post } from "./Post";
import { User } from "./User";

// many to many relationship
// user <-> post
// user -> join table <- post
// user -> updoot <- post

@Entity() // typeorm
export class Updoot extends BaseEntity {
  @Column({ type: "int" })
  value: number;

  @PrimaryColumn()
  userId: number;

  @ManyToOne(() => User, (user) => user.updoots)
  user: User;

  @PrimaryColumn()
  postId: number;

  @ManyToOne(() => Post, (post) => post.updoots)
  post: Post;
}
```

Update index.ts

- We have to tell typeorm to use this Updoot entity , by adding it to `entities: [Post, User],` in the typeorm config passed into createConnection()

/index.ts

```
const conn = await createConnection({
  type: "postgres",
  database: "lireddit2",
  username: "postgres",
  password: "postgres",
  logging: true,
  synchronize: true,
  migrations: [path.join(__dirname, "./migrations/*")],
  entities: [Post, User, Updoot],
});
```

Update User entity

- Add tthe Updoots to User entity
- Note that we're not exposing the updoots to the client with a @Field() attribute

/entities/User.ts

```
@OneToMany(() => Updoot, (updoot) => updoot.user)
  updoots: Updoot[];
```

Update Post entity

- Add tthe Updoots to Post entity
- Also add a points field that will hold the total upvoates of this post
- Note that we're not exposing the updoots to the client with a @Field() attribute

/entities/Post.ts

```
@Field()
@Column({ type: "int", default: 0 })
points!: number;
```

```
@OneToMany(() => Updoot, (updoot) => updoot.post)
updoots: Updoot[];
```