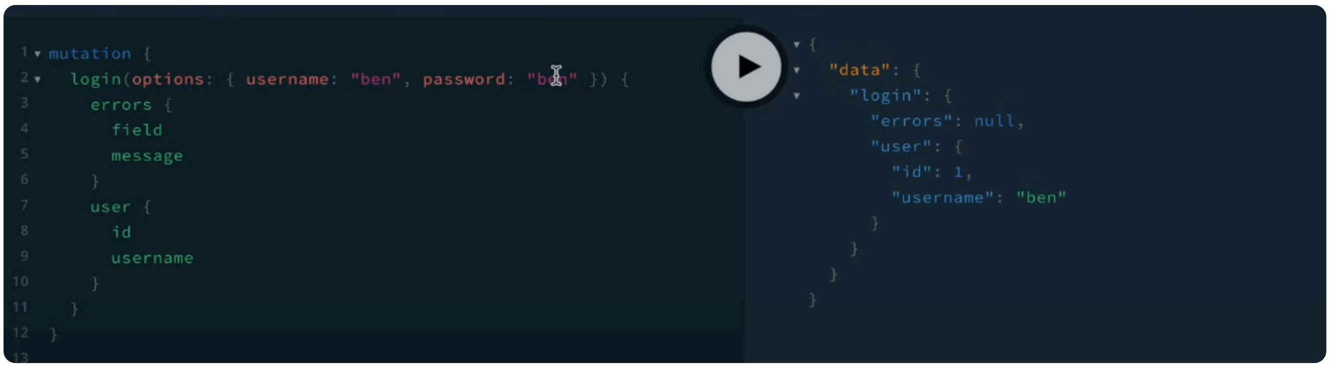
9. Resolver - user.ts / Mutation - login()

#graphql #resolver #authentication #mutation #mikroorm #backend

- Update the user resolver, adding the login() mutation, as shown below:
- An @ObjectType can be defined to be used as return values from mutations and queries
- An @InputType can be defined to be passed into a mutation or query as an input variable

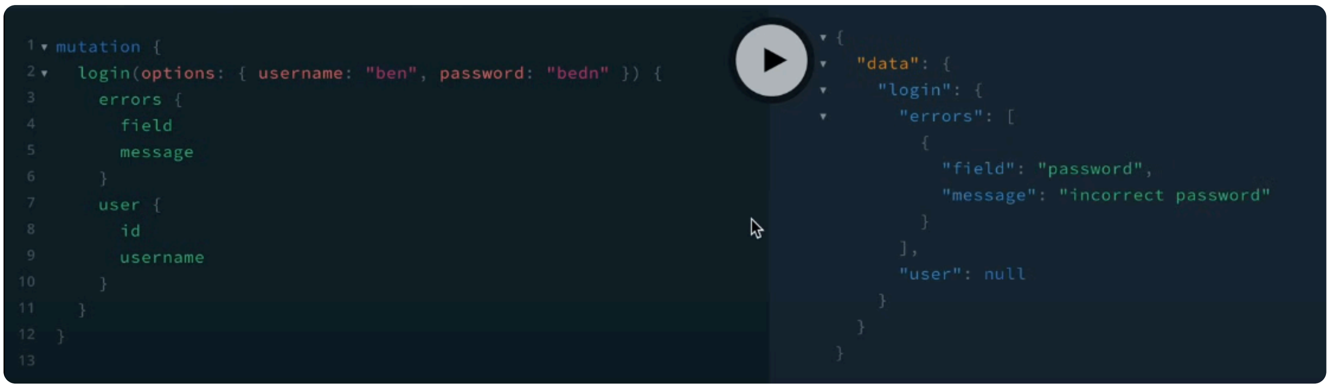
/resolvers/user.ts

```
@ObjectType() // ObjectTypes are returned from Queries and Mutations
class FieldError {
  @Field()
  field: string; // which field the error is about
  @Field()
  message: string; // error message
}

@ObjectType()
class UserResponse {
  @Field(() => [FieldError], { nullable: true })
  errors?: FieldError[];

  @Field(() => User, { nullable: true })
  user?: User;
}



@Resolver()
export class UserResolver {

  @Mutation(() => UserResponse)
  async login(
    @Arg("usernameOrEmail") usernameOrEmail: string,
    @Arg("password") password: string,
    @Ctx() { em, req }: MyContext
  ): Promise<UserResponse> {
    const user = await em.findOne(
      User,
      usernameOrEmail.includes("@")
        ? { email: usernameOrEmail }
        : { username: usernameOrEmail }
    );
    if (!user) {
      return {
        errors: [
          {
            field: "usernameOrEmail",
            message: "That username or email does not exist",
          },
        ],
      };
    }

    const isPasswordValid = await argon2.verify(user.password, password);
    if (!isPasswordValid) {
      return {
        errors: [
          {
            field: "password",
            message: "Incorrect password",
          },
        ],
      };
    }

    req.session.userId = user.id; // created new type for req in types.ts (5) to
make this work, so the session can store the userId

    return { user };
  }

  return user;
  }
}
```



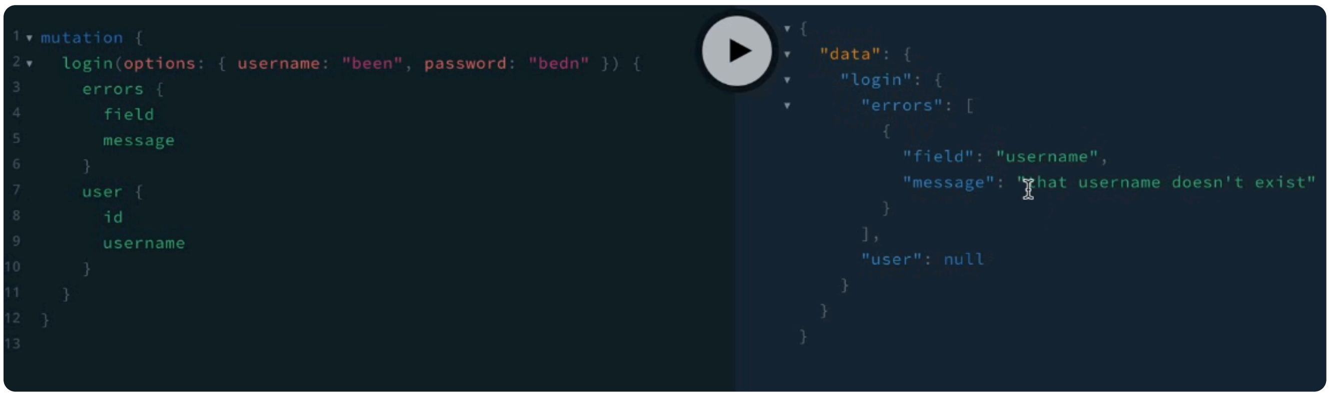
```graphql
mutation {
  login(options: { username: "ben", password: "ben" }) {
    errors {
      field
      message
    }
    user {
      id
      username
    }
  }
}
```

```json
{
  "data": {
    "login": {
      "errors": null,
      "user": {
        "id": 1,
        "username": "ben"
      }
    }
  }
}
```



```graphql
mutation {
  login(options: { username: "ben", password: "bedn" }) {
    errors {
      field
      message
    }
    user {
      id
      username
    }
  }
}
```

```json
{
  "data": {
    "login": {
      "errors": [
        {
          "field": "password",
          "message": "incorrect password"
        }
      ],
      "user": null
    }
  }
}
```



```graphql
mutation {
  login(options: { username: "been", password: "bedn" }) {
    errors {
      field
      message
    }
    user {
      id
      username
    }
  }
}
```

```json
{
  "data": {
    "login": {
      "errors": [
        {
          "field": "username",
          "message": "that username doesn't exist"
        }
      ],
      "user": null
    }
  }
}
```