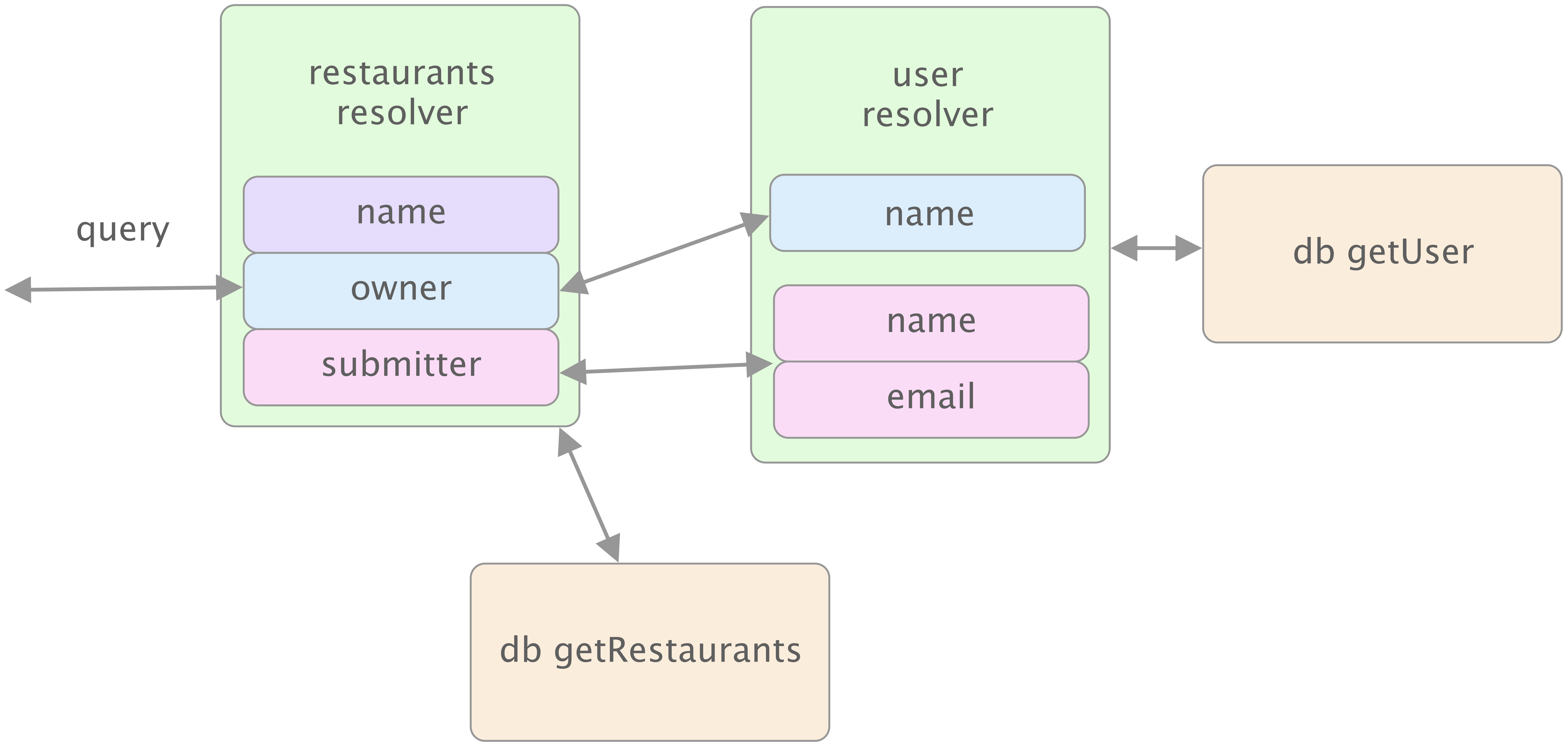
Authorization

How to handle Authorization?

Depends on the app!
Could be all the same authorisation, but mostly has to be decided on case by case basis.

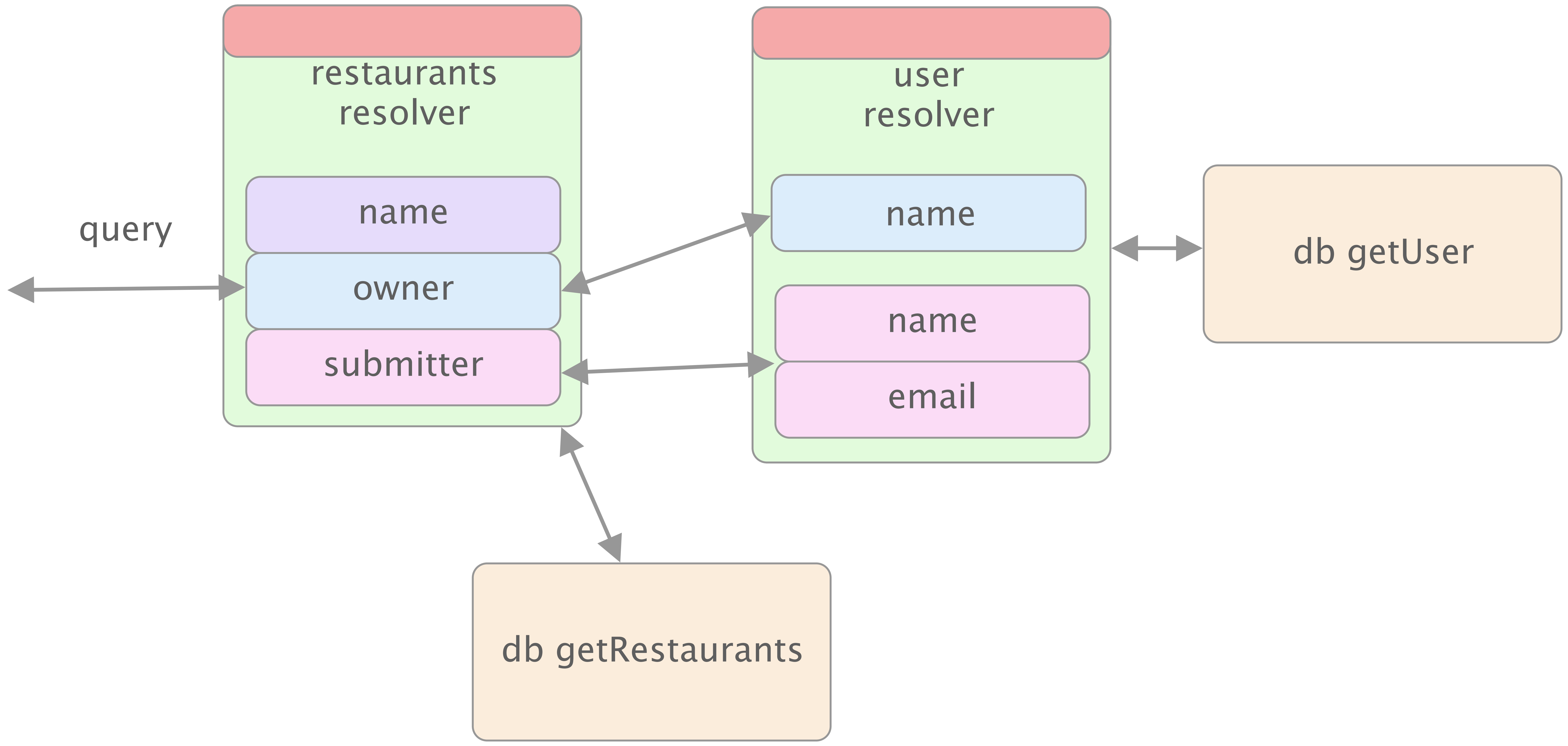
How to handle Authorization?

```
{
  restaurants {
    name
    owner {
      name
    }
    submitter {
      name
      email
    }
  }
}
```



Resolvers

```
{
  restaurants {
    name
    owner {
      name
    }
    submitter {
      name
      email
    }
  }
}
```

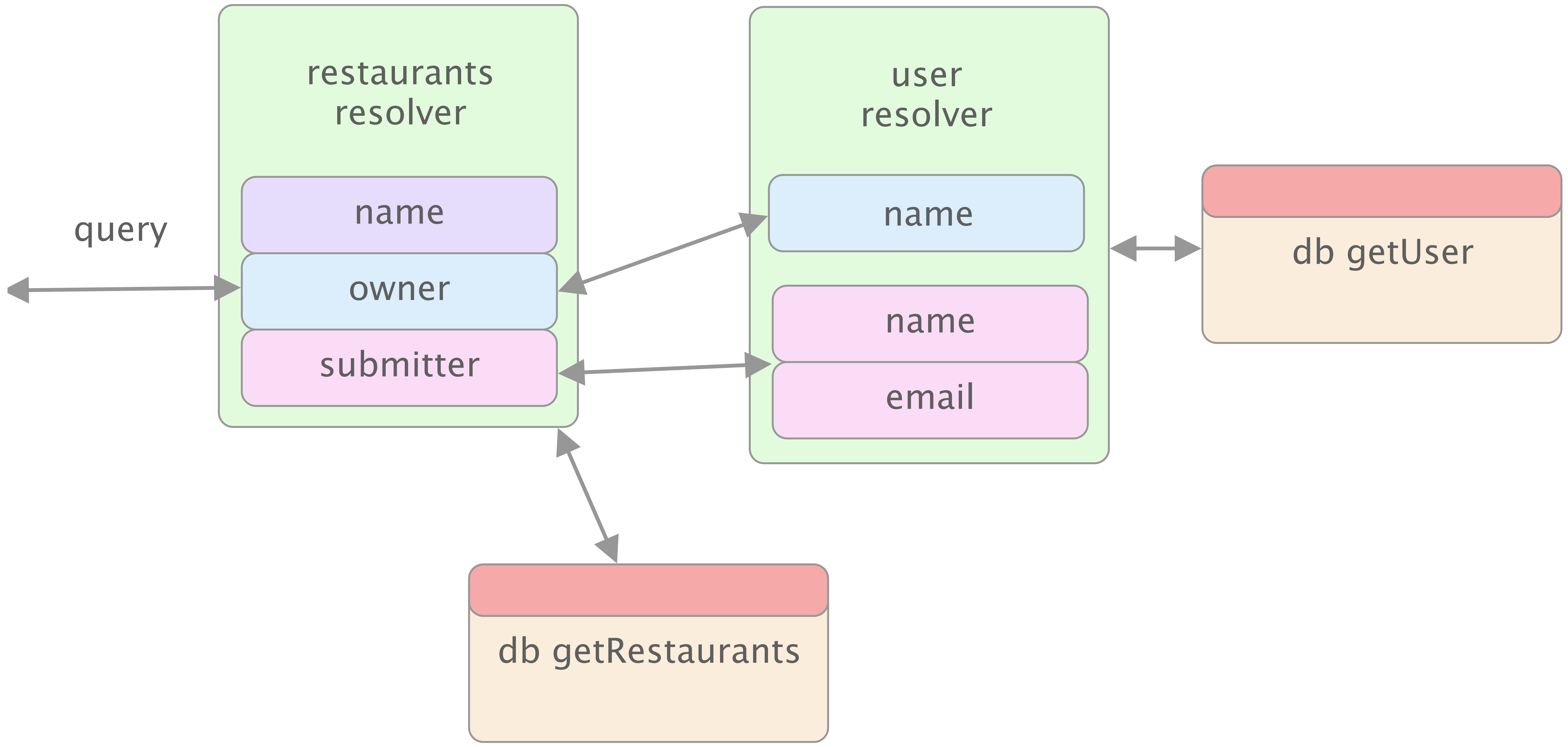


```
const { ForbiddenError } = require("apollo-server")

module.exports = {
  Mutation: {
    addFruitToBasket: (_parent, args, context) => {
      if (!context.user)
        throw new ForbiddenError("Please login first")
      return addFruitToBasket({ …args, user });
    },
  }
};
```

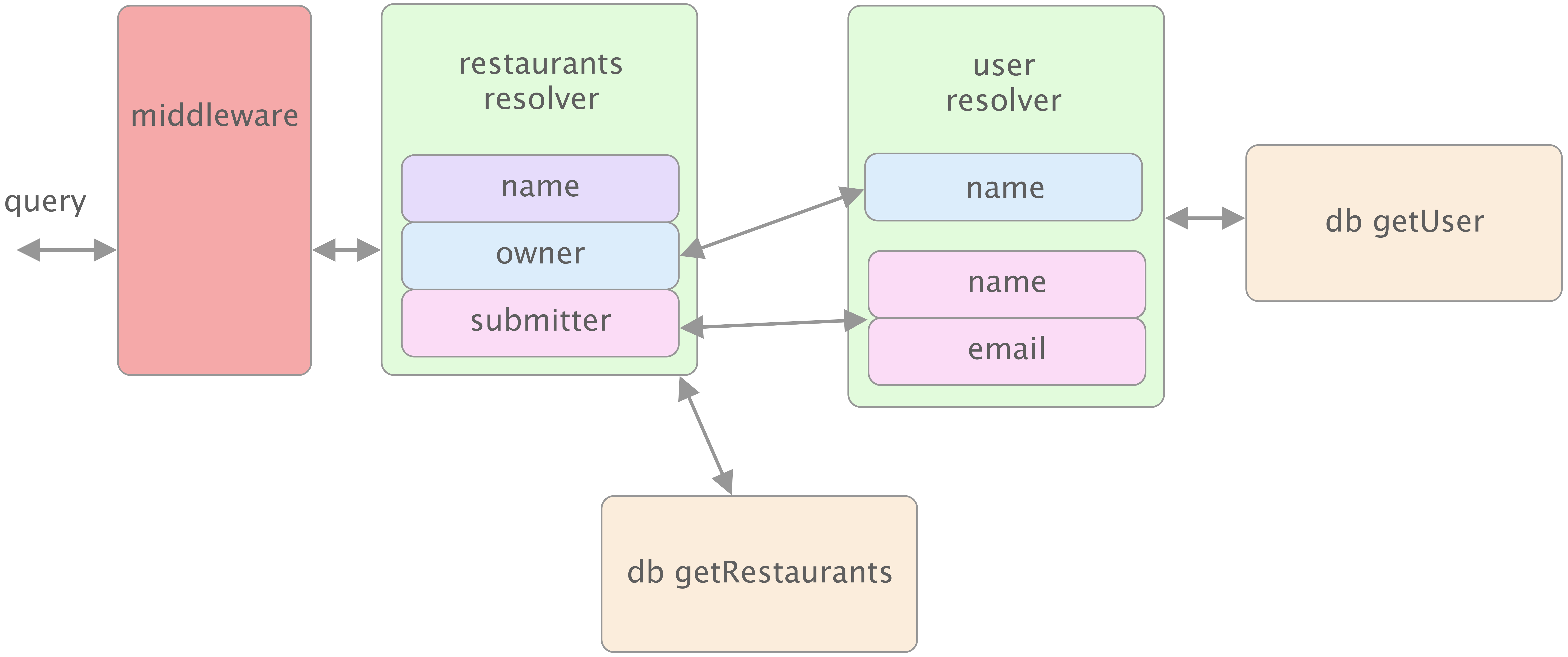
Models/REST Layer

```
{
  restaurants {
    name
    owner {
      name
    }
    submitter {
      name
      email
    }
  }
}
```



Middleware

```
{
  restaurants {
    name
    owner {
      name
    }
    submitter {
      name
      email
    }
  }
}
```



```
const { rule, shield, and, or, not } = require("graphql-shield");
const { applyMiddleware } = require("graphql-middleware");

const isAuthenticated = rule()(async (parent, args, ctx) => ctx.user !== null);
const isAdmin = rule()(async (parent, args, ctx) => ctx.user.role === "admin");
const isEditor = rule()(async (parent, args, ctx) => ctx.user.role === "editor");

const permissions = shield({
  Query: {
    frontPage: not(isAuthenticated),
    fruits: and(isAuthenticated, or(isAdmin, isEditor)),
    customers: and(isAuthenticated, isAdmin)
  },
  Mutation: {
    addFruitToBasket: isAuthenticated
  },
  Fruit: isAuthenticated,
  Customer: isAdmin
});

const schema = applyMiddleware(schema, permissions);
```

Exercise

Secure the createRestaurant mutation so only a logged in user can do the mutation.

Exercise

During creation of the Restaurant add the logged in user as submitter. Make sure only the current user can see the email.