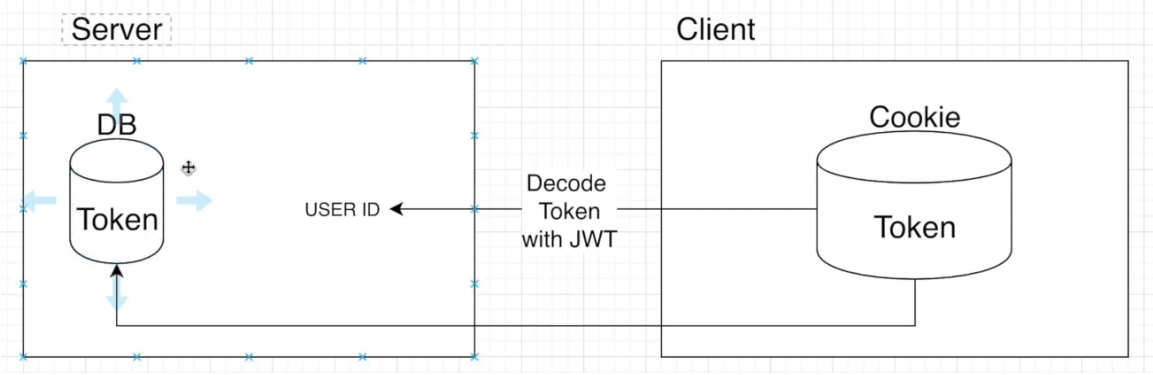
#13 Auth 기능

- 웹 사이트 내 권한 제어를 위해 Auth 기능 필요



- 쿠키에 저장된 Token을 서버에서 가져와 복호화하여 DB값과 비교

→ client의 Cookie에서 토큰을 가져와서 복호화하고, DB값과 비교하는 역할을 수행하는 Middleware를 추가함

```
//boiler-plate/middleware/auth.js
const {User} = require('./models/User') //User모델 불러옴

let auth = (req, res, next)=>{ //인증 처리 수행

    //1. client cookie에서 토큰 가져옴
    let token = req.cookies.x_auth; //쿠키에서 x_auth라는 이름의 필드를 가져옴

    //2. 토큰 복호화한 후, DB에서 유저를 찾음
    User.findByToken(token, function(err, user){
        if(err) throw err;
        if(!user) return res.json({isAuth : false, error : true})

        //request 정보에 세팅 후, middleware 이후 로직 수행
        req.token = token;
        req.user = user;
        next();
    })

    //3. 유저가 있으면 인증 OK

    //4. 유저가 없으면 인증 NO
}
```

```
module.exports = {auth} //다른 파일에서 쓸 수 있또록 export 처리
```

```
//boiler-plate/index.js
//Auth route 생성
app.get('api/users/auth', auth,(req, res)=>{
  //auth middleware 추가 -> endpoint 실행 후 callback 전 수행되는 부분

  //여기가 실행된다면 authentication = true
  console.log('authentification is true')

  //client에게 정상 상태와 user 정보를 전달
  res.status(200)
  .json({
    _id : req.user._id,
    isAdmin : req.user.role == 0 ? false : true,
    isAuth : true,
    email : req.user.email,
    name : req.user.name,
    lastname : req.user.lastname,
    role : req.user.role,
    iamge : req.user.iamge
  })
})
```