AUTH 기능 만들기

이해관계자	
상태	
작성자	
문서 유형	
작성일시	@2022년 7월 10일 오후 10:38
최종 편집일시	@2022년 7월 11일 오전 12:35
최종 편집자	

Client token 에서 서버로 유효한 토큰인지 계속해서 요청

(Cookie 토큰 에 secret key를 decode 하면 USER_ID가 나옴)

1. 쿠키에 저장된 토큰을 서버에서 가져와서 복호화한다.
2. //복호화해서 나온 아이디로 유저를 찾은 후 해당 유저가 토큰을 보유하고 있는지 확인

```
//미들웨어 auth생성 필요
app.get('/api/users/auth', auth ,(req,res) => {
  //여기까지 미들웨어를 통과해 왔다면 auth가 true
  res.status(200).json({
    _id :req.user._id,
    isAdmin : req.user.role == 0 ? false : true,
    isAuth : true,
    email : req.user.emai,
    name:req.user.name,
    lastname : req.user.lastname,
    role: req.user.role,
    image : req.user.image
  })

})
```

```
//auth.js
const {User} = require("../models/User")

let auth = (req, res, next) => {
    //인증처리

    // 클라이언트 쿠키에서 토큰을 가져온다.
    let token = req.cookie.x_auth;

    // 토큰을 복호화 한 후 유저를 찾는다.
```

```
    User.findByToken(token, (err, user) =>{
        if(err) throw err;
        if(!user) return res.json({isAuth:false, error:true})

        req.token = token;
        req.user = user;
        next(); //next가 없으면 middleware에서 빠져나갈수가 없다
    })
}

module.exports = { auth };
```

```
userSchema.statics.findByToken = function(token, cb){
    let user = this;
    //토큰을 디코드 한다.
    jwt.verify(token, 'secretToken', function(err, decoded){
        //docode 된 값은 아이디
        //클라이언트에서 가져온 토큰과 디비에 보관된 토큰이 일치하는지 확인
        user.findOne({"_id":decoded, "token" :token }, function(err, user){
            if(err) return cb(err);
            cb(null, user);
        })
    })
}
```