#11 #12 로그인과 토큰생성

- 로그인 route 생성
  1. DB에 요청된 이메일 찾기
  2. 이메일의 비밀번호가 같은지 확인

```
//비밀번호 비교하는 함수 선언

//선언 : boiler-plate/User.js
userSchema.comparePassword = function(plainPassword, callbacnkFunc(){
  bcrypt.compare(plainPassword, this.password, function(err, isMatch){
    if(err) return callbackFunc(err)
    callback(null, isMatch)
  })
});
```

```
//호출 : boiler-plate/index.js
user.comparePassword(req.body.password, (err, isMatch)=>{
  //비밀번호 불일치
  if(!isMatch) return json({loginSucess : false, message : "비밀번호 틀림"})

  //비밀번호 일치하면 토큰 생성
});
```

3. 비밀번호 일치 시 토큰 생성

```
//토큰 생성 함수 선언

//선언 : boiler-plate/User.js

const jwt = require('jsonwebtoken')

userSchema.methods.generateToken = function(callbackFunc(){
  var user = this;

  //user._id + secretToken 텍스트를 이용해 토큰 생성

  var token = jwt.sign(user._id.toHexString(), 'secretKey')
  //토큰 생성 시 id값 hexstring()으로 변환
  user.token = token //생성된 토큰을 모델에 저장

  user.save(function(err, user){
    if(err) return callbackFunc(err)
    callbackFunc(null, user)
```

```
  })
})
```

```
//boiler-plate/index.js
user.generate((err, user)=>{
  if(err) res.status(400).send(err)

  res.cookie("x_auth", user.token) //cookie에 x_auth란 이름으로 토큰 저장
  .status(200)
  .json({login:seccess : true, userId : user._id})
})
```

- 토큰생성 : JSONWEBTOKEN 라이브러리 필요
  - https://www.npmjs.com/package/jsonwebtoken
  - 전자서명된 url-safe(URL로 이용할 수 있는 문자만 구성된) JSON
  - JWT 사용 : 로그인 시 서버는 유저의 정보에 기반한 토큰을 발급해 유저에게 전달 → 유저가 서버에 요청할 때 마다 JWT를 포함하여 전달하면 서버는 요청을 받을 때 마다 해당 토큰의 유효성, 인증, 권한 등을 확인하여 작업을 처리함 → 유저의 세션을 유지할 필요 없이 토큰만 확인할 수 있어 세션관리 필요 없음 → 자원 절약

```
npm install jsonwebtoken --save
```

- 토큰 쿠키에 저장하기 위한 cookieparser 모듈 설치
  - cookie-parser : 요청된 쿠키를 쉽게 추출할 수 있도록 도와주는 미들웨어로 express의 request 객체에 cookies 속성이 부여됨
  - 설치

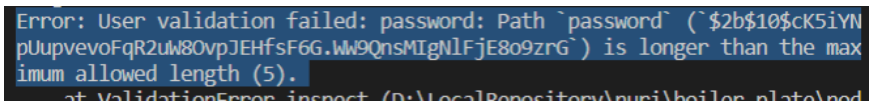
```
npm install cookie-parser --save
```

  - 생성과 삭제

```
const cookieParser = require('cookie-parser');

app.use(cookieParser());

//생성
res.cookie('hasVisited', '1', {
    maxAge: 60*60*1000,
    httpOnly: true,
```

```
    path:'/visitors'
});


//삭제
res.clearCookie('hasVisited', {path: '/visitors'})
```

- 수정사항



User.js 스키마에서 password 사이즈를 5로 설정했었는데 암호화된 값을 가져오면서 password 사이즈가 길어져 user 모델에 담을 때 에러 발생 → password maxlength 100 으로 변경 후 에러 개선