


SeSAC 4기, 

웹 풀스택 과정 passport 수업

WITH 팀 리처드



passport

Passport란?

- 사용자 인증 절차에 대한 로직을 간단하게 구현할 수 있도록 도와주는 **미들웨어**
- 복잡한 세션과 쿠키를 안전하고 간단하게 설계하도록 도와준다.
- Strategy(전략) 을 사용해 사용자 인증
- Strategy란?
 - 어떤 것을 이용해, 어떻게 인증을 구현할 것인가에 대한 전략
 - passport-local, passport-kakao, passport-google-oauth 등
- [Passport.js \(passportjs.org\)](https://passportjs.org)

Passport 큰 흐름 (1)

- 로그인 과정

1. 로그인 요청
2. passport.authenticate 메서드 호출
3. 로그인 전략 수행
4. 로그인 성공 시 사용자 정보 객체와 함께 req.login 호출
5. req.login 메서드가 passport.serializeUser의 콜백함수 호출
6. req.session에 사용자 아이디만 저장
7. 로그인 완료

Passport 큰 흐름 (2)

- 로그인 이후 과정
 1. 모든 요청에 대해 `passport.deserializeUser` 메서드 호출
 2. `Req.session`에 저장된 아이디로 데이터베이스에서 사용자 조회
 3. 조회된 사용자 정보를 `req.user`에 저장
 4. 라우터에서 `req.user` 객체 사용 가능

Passport 설치

```
npm i passport passport-local express-session cookie-parser
```

```
license: MIT  
"dependencies": {  
  "cookie-parser": "^1.4.6",  
  "express": "^4.18.1",  
  "express-session": "^1.17.3",  
  "passport": "^0.6.0",  
  "passport-local": "^1.0.0"  
}
```

Passport 사용하기

```
const passport = require('passport');  
app.use(passport.initialize());  
app.use(passport.session());
```

app이 passport를
사용하도록 초기화

- passport가 app에서 설정한 express session을 사용하도록 설정
- 세션을 사용하기 때문에 express-session 미들웨어 다음에 위치시켜야 한다.

Passport 사용하기

```
const LocalStrategy = require('passport-local').Strategy;  
passport.use(  
  new LocalStrategy((username, password, done) => {  
  })  
)
```



- Username, password가 서버에 저장된 정보와 일치하는지 확인 후 로그인 승인

Passport 사용하기

- 사용자 인증

```
app.post('/login', passport.authenticate('local', {  
  successRedirect: '/',  
  failureRedirect: '/login',  
}));
```



지정 전략(strategy)를 사용해 로그인에 성공/실패할 경우 이동할 경로와 메시지 설정하기

Passport 사용하기

- 세션 기록과 읽기

- Passport.authenticate 에서 로그인 성공 시 실행할 콜백함수 등록
- 로그인 시 serializeUser에 등록된 콜백함수 실행 -> 서버 세션 (req.session)에 사용자의 식별자 저장



```
passport.serializeUser((user, done) => {});  
passport.deserializeUser((id, done) => {});
```



- 서버로 클라이언트 요청이 있을 때마다 호출되는 콜백함수 등록
- 로그인 성공 후 서버에 요청할 때마다 deserializeUser에 등록된 콜백함수 실행 -> req.session의 사용자 식별자를 이용해 실제 세션의 사용자 정보를 읽어 req.user에 저장

Passport-local 이용하기

- 함께 제공된 nodejs-passport 참고하기!

Passport-google 이용하기

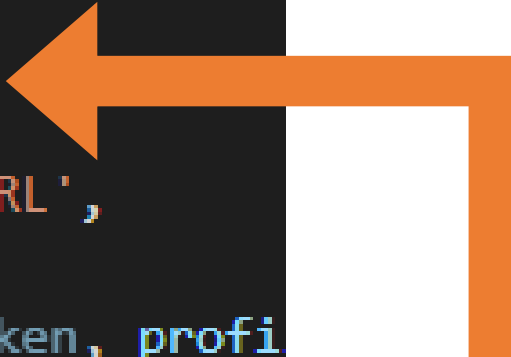
```
npm i passport-google-oauth
```

Passport-google 이용하기

```
module.exports = new GoogleStrategy(  
  {  
    clientID: "구글 ID",  
    clientSecret: "구글비밀키",  
    callbackURL: '등록한 콜백 URL',  
  },  
  async (accessToken, refreshToken, profile)
```

OAuth 2.0 클라이언트 ID

<input type="checkbox"/>	이름	생성일 ↓	유형	클라이언트 ID	작업
<input type="checkbox"/>	_____	2022. 6. 5.	웹 애플리케이션	53805674361-h7n96... 	  



DATE _____ **TIME** _____

Passport-google 이용하기

```
module.exports = new GoogleStrategy(  
  {  
    clientID: "구글 ID",  
    clientSecret: "구글비밀키",  
    callbackURL: '등록한 콜백 URL',  
  },  
  async (accessToken, refreshToken)
```

```
router.get(  
  'google/callback',  
  passport.authenticate('google', {  
    failureRedirect: '/',  
  }),  
  (req, res) => {  
    res.redirect('/');  
  }  
);
```

승인된 리디렉션 URI ?

웹 서버의 요청에 사용

URI 1 *
http://localhost:3001/google/callback

+ URI 추가

Passport-kakao 이용하기

```
npm i passport-kakao
```


Passport-kakao 이용하기

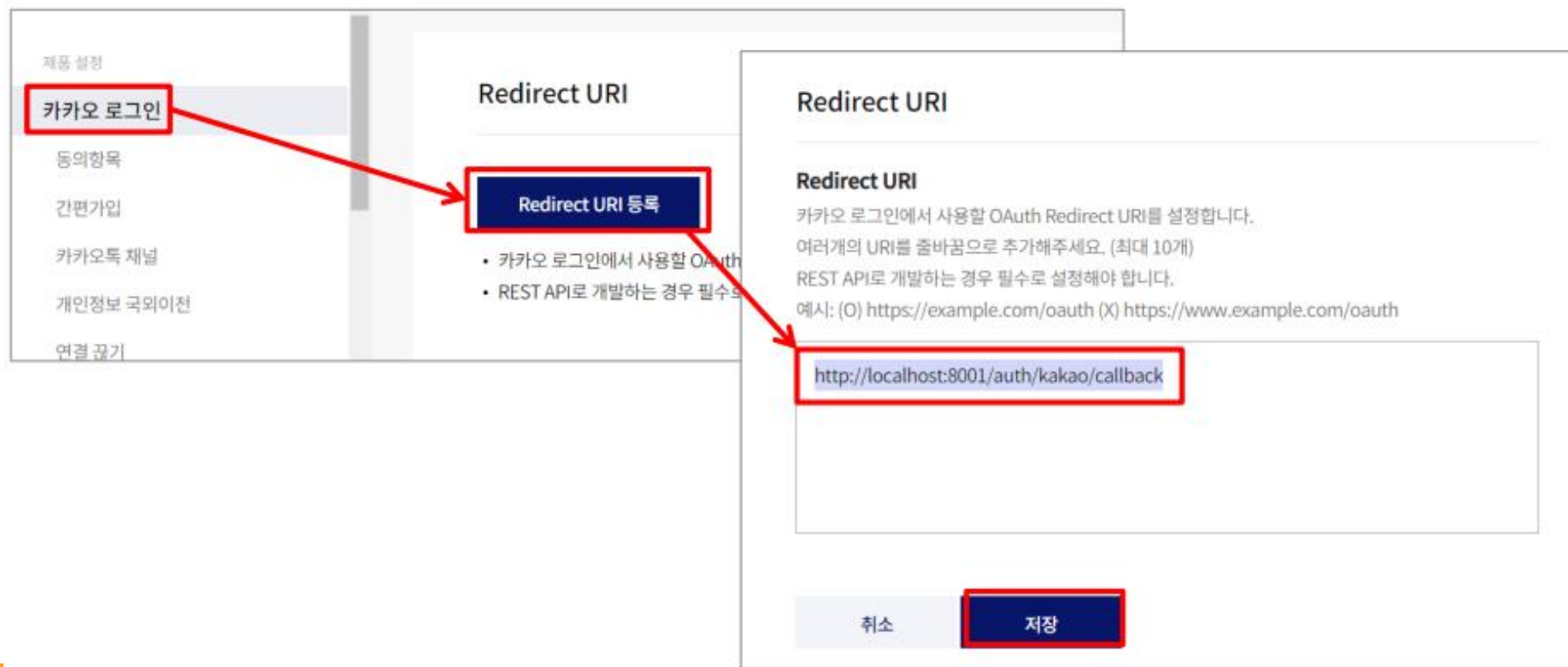
```
module.exports = new KakaoStrategy(  
  {  
    clientID: '카카오 ID',  
    callbackURL: '카카오 콜백 URL',  
  },  
);
```

앱 키

네이티브 앱 키	4e05a4049f543586fe8d841b26cbbb8f
REST API 키	ab262a2d8b8d2c85f8919f40a7e80214
JavaScript 키	fc570c96f19dc75c8a86d8ec1037eeb8

Passport-kakao 이용하기

```
module.exports = new KakaoStrategy(  
  {  
    clientID: '카카오 ID',  
    callbackURL: '카카오 콜백 URL',  
  }  
);
```



The image shows the Kakao Developer Console interface. On the left, a sidebar menu has '카카오 로그인' (Kakao Login) highlighted. A red arrow points from this menu item to the 'Redirect URI 등록' (Register Redirect URI) button in the main area. Below this button, there are instructions in Korean and two bullet points: '카카오 로그인에서 사용할 OAuth' and 'REST API로 개발하는 경우 필수로'. Another red arrow points from the 'Redirect URI 등록' button to a text input field containing the URL 'http://localhost:8001/auth/kakao/callback'. At the bottom right, there are two buttons: '취소' (Cancel) and '저장' (Save), with '저장' being highlighted.

제품 설정

- 카카오 로그인
- 동의항목
- 간편가입
- 카카오톡 채널
- 개인정보 국외이전
- 연결 끊기

Redirect URI

Redirect URI 등록

- 카카오 로그인에서 사용할 OAuth
- REST API로 개발하는 경우 필수로

Redirect URI

카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다.
여러개의 URI를 줄바꿈으로 추가해주세요. (최대 10개)
REST API로 개발하는 경우 필수로 설정해야 합니다.
예시: (O) https://example.com/oauth (X) https://www.example.com/oauth

http://localhost:8001/auth/kakao/callback

취소 저장

DB를 연결했다면?

```
passport > JS kakaoStrategy.js > [?] <unknown> > [?] <function>
...
1 const KakaoStrategy = require('passport-kakao').Strategy;
2 const { User, Point } = require('../models');
3
4 module.exports = new KakaoStrategy(
5   {
6     clientID: process.env.KAKAO_ID,
7     callbackURL: '/auth/kakao/callback',
8   },
9   async (accessToken, refreshToken, profile, done) => {
10     console.info('__new KakaoStrategy()');
11     try {
12       const exUser = await User.findOne({
13         where: { sns_id: profile.id, provider: 'kakao' },
14       });
15       if (exUser) {
16         console.log('__kakao exUser', exUser);
17         done(null, exUser);
18       } else {
19         const newUser = await User.create({
20           email: profile._json && profile._json.kakao_account.email,
21           nickname: profile.displayName,
22           sns_id: profile.id,
23           provider: 'kakao',
24         });
25         done(null, newUser);
26       }
27     } catch (error) {
28       console.error(error);
29       done(error);
30     }
31   }
32 );
```

```
passport > JS googleStrategy.js > [?] <unknown> > [?] <function>
...
1 const GoogleStrategy = require('passport-google-oauth').OAuth2Strategy;
2 const { User } = require('../models');
3
4 module.exports = new GoogleStrategy(
5   {
6     clientID: process.env.GOOGLE_ID,
7     clientSecret: process.env.GOOGLE_SECRET,
8     callbackURL: '/auth/google/callback',
9   },
10  async (accessToken, refreshToken, profile, done) => {
11    console.info('__new GoogleStrategy()');
12    console.log('__google profile', profile);
13    try {
14      const exUser = await User.findOne({
15        where: { sns_id: profile.id, provider: 'google' },
16      });
17      if (exUser) {
18        console.log('__google exUser', exUser);
19        done(null, exUser);
20      } else {
21        const newUser = await User.create({
22          email: profile._json && profile._json.email,
23          nickname: profile.displayName,
24          sns_id: profile.id,
25          provider: 'google',
26        });
27        done(null, newUser);
28      }
29    } catch (error) {
30      console.error(error);
31      done(error);
32    }
33  }
34 );
```

DB를 연결했다면?

```
passport > JS index.js > ...
You, 4개월 전 | 1 author (You)
1  const passport = require('passport');
2  const kakaoStrategy = require('./kakaoStrategy');
3  const localStrategy = require('./localStrategy');
4  const googleStrategy = require('./googleStrategy');
5  const { User } = require('../models');
6
7  passport.serializeUser((user, done) => {
8    console.info('__passport.serializeUser()');
9    done(null, user.user_id);
10 });
11
12 passport.deserializeUser((user_id, done) => {
13   console.info('__passport.deserializeUser()');
14   User.findOne({ where: { user_id } })
15     .then((user) => done(null, user))
16     .catch((err) => done(err));
17 });
18
19 passport.use(localStrategy);
20 passport.use(kakaoStrategy);
21 passport.use(googleStrategy);
22
23 module.exports = passport;
You, 4개월 전 • fea
```