- cookie
  - 참조 : [Nodejs] 쿠키(Cookie) 사용하기
- 보안 이슈

Cookie란

- 쿠키는 서버 측에서 클라이언트에게 보내주는 '키-밸류' 형식의 데이터입니다. 이 데이터는 클라이언트가 가지고 있다가 서버에 다시 접속 요청을 할 때 서버에게 전송합니다.
- 쿠키는 암호화가 되지 않은 데이터이므로 절대 안전하지 않습니다. 따라서 Password와 같은 기밀정보들은 쿠키로 전송해서는 안됩니다. http 프로토콜로 전송되는 패킷들은 아주아주 손쉽게 훔칠 수 있기 때문입니다.
- 또한, 최근에 들어서는 쿠키의 사이즈가 4KB밖에 안된다는 점. 매 요청마다 쿠키를 전송한다는 점 때문에 사용을 지양하는 편입니다. 따라서 대체 기술인 IndexedDB, 웹 스토리지 등을 사용

Cookie 용도

- 세션 관리(Session management)
  - 로그인 유지, 장바구니 유지, 게임 스코어 관리하는 용도
- 개인화(Personallization)
  - 사용자의 선호 언어, 테마 등을 유지하는 용도
-
- 트래킹(Tracking)
  - 사용자의 방문통계 등 행동을 기록하고 분석하기 위한 용도

node.js에서 쿠기 저장

```
var http = require('http');
http.createServer(function(request, response){
    response.writeHead(200, {
        'Set-Cookie':['yummy_cookie=choco', 'tasty_cookie=strawberry']
    });
    response.end('Cookie!!');
}).listen(3000);
```

```
  res.writeHead(200, {
    "Set-Cookie": [
      "yummy_cookie=choco",
      "tasty_cookie=strawberry",
      `Permanent=cookies; Max-Age=${60 * 60 * 24 * 30}`,
      "Secure=Secure; Secure",
      "HttpOnly=HttpOnly; HttpOnly",
      "Path=Path; Path=/cookie",
      "Doamin=Domain; Domain=app.sample.com",
    ],
  });
```

쿠키 읽기

```
const http = require("http");
const cookie = require("cookie");
const server = http.createServer(function (request, response) {
  console.log(request.headers.cookie);
  var cookies = {};
  if (request.headers.cookie !== undefined) {
    cookies = cookie.parse(request.headers.cookie);
  }
  console.log(cookies.yummy_cookie);
  response.writeHead(200);
  response.end("hello");
});
server.listen(3000, function () {
  console.log("server runtime http://localhost:3000");
});
```

보안이슈

- 취약한 종속성 관리
  - 보안 취약점을 가지고 있는 외부 패키지
  - npm 업데이트, 보안 업데이트 수시로 확인
  - Helmet 보안모듈
- 인증과 세션관리
  - 사용자인증에 암호화 알고리즘(패스워드 암호화) : passport, bcrypt모듈
  - 세션식별자는 쿠키나 JWT(json Web token)활용
  - 세션 타임아웃
- XSS(크로스사이트스크립팅) 공격
  - 입력값 검증. 입력값(<,') 이스케이프 처리
  - Content Security Policy(CSP)설정하여 스크립트 실행, 외부 리소스 로딩 제한
- 보안 감사 로깅
  - 중요한 이벤트 기록
- nginx 서버구축
  - 무중단서버
- 커스텀 에러페이지 작성

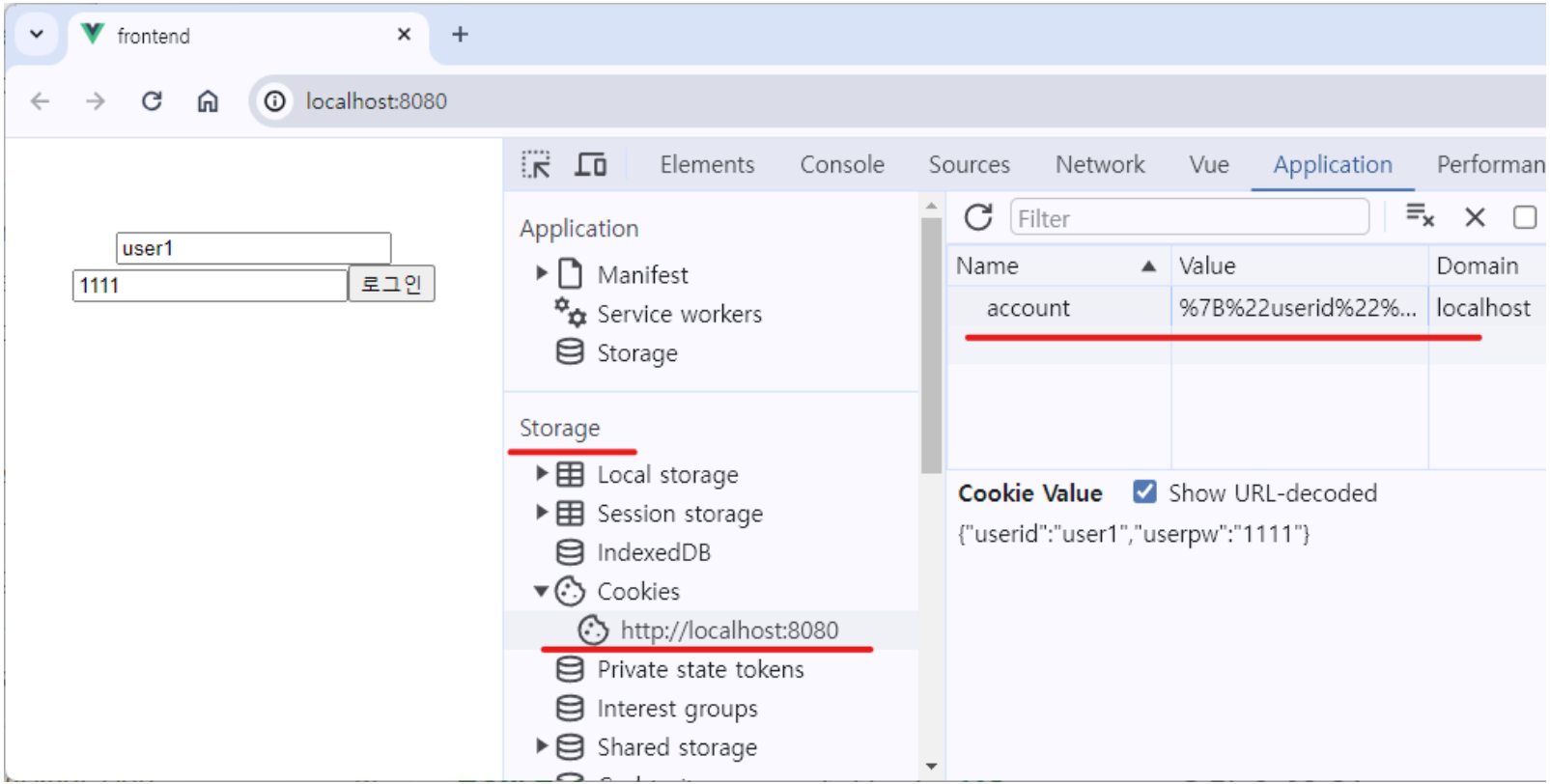
```
const express = require('express');
const session = require('express-session');
const fileStore = require('session-file-store')(session);
const app = express();

app.use(session({
  secret: 'secret key', //암호화하는 데 쓰일 키
  resave: false,        // 세션에 변경사항이 없어도 항상 다시 저장할지 여부
  saveUninitialized: true,  // 초기화되지 않은 세션을 스토어(저장소)에 강제로 저장할지 여부
  cookie: {           // 세션 쿠키 설정 (세션 관리 시 클라이언트에 보내는 쿠키)
    httpOnly: true,  // true 이면 클라이언트 자바스크립트에서 document.cookie로 쿠키 정보를 볼 수 없음
    secure: true,    // true 이면 https 환경에서만 쿠키 정보를 주고 받도록 처리,
    maxAge: 60000    // 쿠키가 유지되는 시간 (밀리세컨드 단위)
  },
  store: new fileStore() // 세션 저장소로 fileStore 사용
}));
```

- 로그인 요청시 사용자 정보 확인 후 세션에 사용자 정보 저장

```
app.post('/login', (req, res, next) => {
  const {email, pw} = req.body.param;
  // 데이터베이스의 사용자 테이블에서 로그인 인증 처리 코드 작성
  // 사용자가 존재하면(로그인 처리가 성공하면)
  req.session.email = email;      // 세션에 사용자 이메일 정보 저장
  req.session.is_logined = true; // 세션에 로그인 여부 저장
  req.session.save(err => {       // 세션 저장
    if(err) throw err;
    res.redirect('/home');        // 로그인 후 홈화면으로 이동
  });
});
```

- 로그아웃 요청시 세션 삭제 후 로그인 페이지로 이동

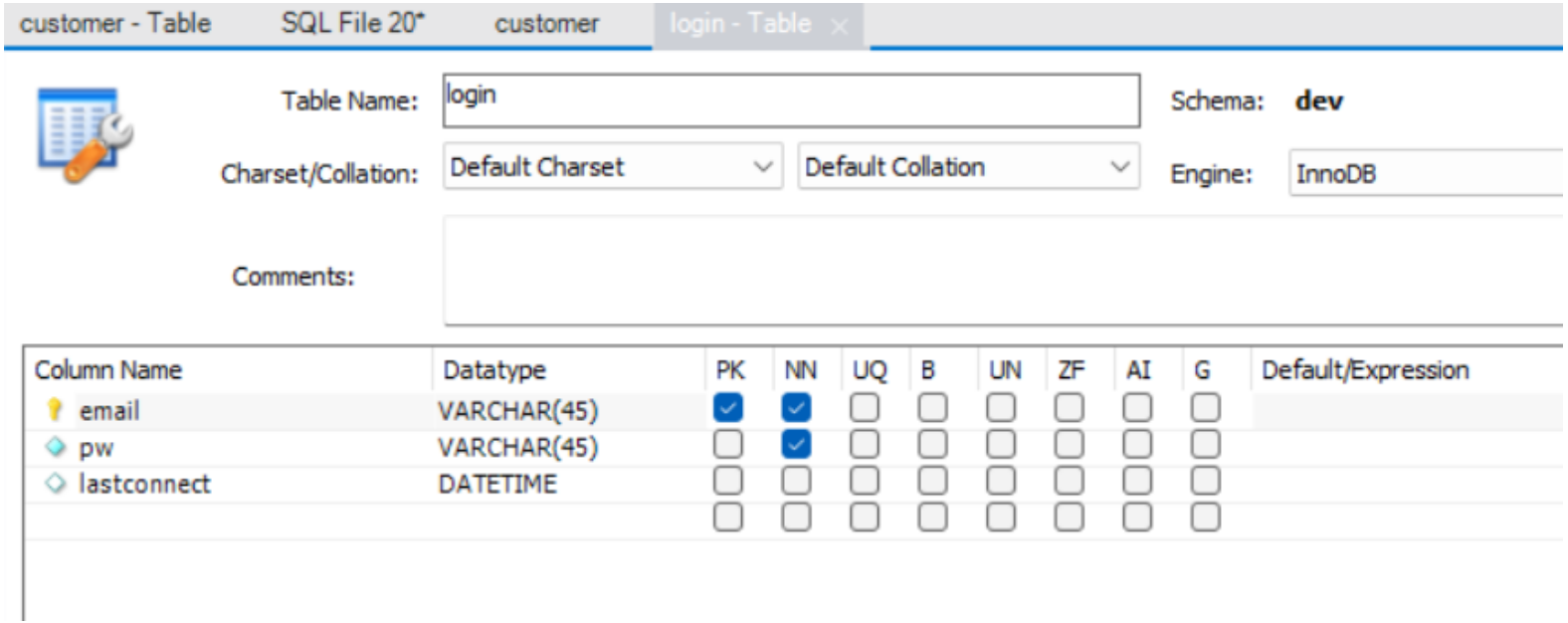
```
app.post('/logout', (req, res, next) => {
  req.session.destroy();         // destroy() 함수를 사용해서 세션 삭제
  res.redirect('/login');        // 로그인 페이지로 이동
});
```


frontend
localhost:8080
user1
1111
로그인
Elements
Console
Sources
Network
Vue
Application
Performan
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
IndexedDB
Cookies
http://localhost:8080
Private state tokens
Interest groups
Shared storage
Filter
Name
Value
Domain
account
%7B%22userid%22%...
localhost
Cookie Value
Show URL-decoded
{"userid":"user1","userpw":"1111"}

jwt

- JSON Web Token
- JWT는 서버와 클라이언트 간 정보를 주고 받을 때 Http 리퀘스트 헤더에 JSON 토큰을 넣은 후 서버는 별도의 인증 과정없이 헤더에 포함되어 있는 JWT 정보를 통해 인증

로그인 테이블 생성


customer - Table
SQL File 20*
customer
login - Table
Table Name:
login
Schema:
dev
Charset/Collation:
Default Charset
Default Collation
Engine:
InnoDB
Comments:
Column Name
Datatype
PK
NN
UQ
B
UN
ZF
AI
G
Default/Expression
email
VARCHAR(45)
pw
VARCHAR(45)
lastconnect
DATETIME