



Prijava i registracija

Web programiranje

Jurica Maltar

LocalStorage (JavaScript na klijentu)



- Vrijednosti u pregledniku pohranjene trajno
- `window.localStorage.getItem("someItem");`
- `window.localStorage.setItem("someItem", 10);`
- `window.localStorage.removeItem("someItem");`
- `window.localStorage.clear();`

Session storage (JavaScript na klijentu)



- Vrijednosti pohranjene sve dok se preglednik ne ugasi
- `window.sessionStorage.getItem("someItem");`
- `window.sessionStorage.setItem("someItem", 10);`
- `window.sessionStorage.removeItem("someItem");`
- `window.sessionStorage.clear();`

Lozinka



- Lozinku nije preporučljivo spremiti kao plain-text. Zašto?
- Koristimo built-in biblioteku `crypto` za hash algoritme

```
const randomBytesAsync = util.promisify(crypto.randomBytes);
const pbkdf2Async = util.promisify(crypto.pbkdf2);

const salt = (await randomBytesAsync(16)).toString("hex");
const hash = (
    await pbkdf2Async(password, salt, 1000, 64, "sha512")
).toString("hex");
```



JSON Web Token (abbr. JWT)

- npm install jsonwebtoken
- Token je:
 - nemoguće stvoriti ukoliko vrijednost `secret` nije poznata
 - valjan ako nije istekao i generiran je s valjanom vrijednošću `secret`
- Stvaranje: `jwt.sign`
- Provjera valjanosti: `jwt.verify`



JSON Web Token (abbr. JWT)

- Tijek rada:
 1. Poslužitelj jedini zna **secret**
 2. Pri prijavi/registraciji klijent uz korisničko ime/email pruža lozinku
 - a) Prilikom registracije, hash lozinke spremi se u bazu podataka
 - b) Prilikom prijave, hash lozinke uspoređuje se s hashom lozinke pohranjenim u bazi podataka
 3. Pri uspješnoj prijavi/registraciji, poslužitelj izdaje valjan token i šalje ga klijentu
 4. Klijent pohranjuje token u local storage i koristi ga po potrebi na zaštićenim rutama poslužitelja
 5. Na zaštićenim rutama poslužitelja, provjerava se je li token valjan



JSON Web Token (abbr. JWT)

- Sadržaj tokena:
 - Zaglavlje (header)
 - Payload
 - Potpis (signature)

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJfaWQiOiI1Y2U10WJ1YThiMzU5MTM2M2M1MTAxNDYi  
LCJlbWFpbCI6ImptYWx0YXJAbWF0aG9zLmhyIiwidXNl  
cm5hbWUiOiJqbWFsdGFyIiwiZ3Jhdml0YXJVcmwiOiIv  
L3d3dy5ncmF2YXRhcj5jb20vYXZhGFyL2RhNDkyYzgy  
ODZhNDA4ZTBkZjFiMGFkY2I2YWQ4MWQzMjIwZXhwIjox  
NTU5MTk2MzEzLCJpYXQiOjE1NTg1OTE1MTN9.  
1vE9T-NwxBL5q7wY5YShkAu31MuKvoEjdGtq0SnTo
```

JWT (parsiranje na klijentu)



```
const token = `  
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJfaWQiOiI1Y2U10WJ1YThiMzU5MTM2M2M1MTAxNDYiLCJlbWFpbCI6ImptYWx0YXJAbWF0aG9zLmhyIi  
widXNlcm5hbWUiOiJqbWFsdGFyIiwiZ3JhdmF0YXJVcmwiOiIvL3d3dy5ncmF2YXRhcj5jb20vYXZhdGFy  
L2RhNDkyYzgyODZhNDA4ZTBkZjFiMGFkY2I2YWQ4MWQzMjIwZXhwIjoxNTU5MTk2MzEzLCJpYXQiOjE1NT  
g1OTE1MTN9.  
1vE9T-NwxB_L5q7wY5YShkAu31MuKv_oEjdGtqOSnTo  
`
```

```
const header = JSON.parse(window.atob(token.split(".")[0]));  
const payload = JSON.parse(window.atob(token.split(".")[1]));  
const signature = JSON.parse(window.atob(token.split(".")[2]));  
  
let items = { header, payload, signature };  
  
for(let item in items)  
    console.log(items[item]);
```

JWT (parsiranje na klijentu)



```
{"alg": "HS256", "typ": "JWT"}
```

```
{"_id": "5ce59bea8b3591363c510146", "email": "jmaltar@mathos.hr", "username": "jmaltar", "gravatarUrl": "//www.gravatar.com/avatar/da492c8286a408e0df1b0adcb6ad81d3", "exp": 1559196313, "iat": 1558591513}
```

Öñ=OãpÄæ®õc@.ßS.*ÿè7F¶£■