Baxış Tahir oğlu CƏFƏROV
Qərbi Kaspi Universitetinin magistrantı
E-mail: jafarovbakhish@gmail.com

VEB-TƏTBİQETMƏLƏRDƏ TOKEN-ƏSASLI AUTENTİFİKASİYA VƏ ROL ƏSASLI GİRİŞ NƏZARƏTİ: JWT STANDARTININ TƏHLİLİ VƏ TƏTBİQİ

Xülasə

Bu məqalədə müasir çoxistifadəçili veb-tətbiqlərdə istifadəçi kimliyinin təsdiqlənməsi ilə bağlı əsas problem - autentifikasiya mexanizminin seçilməsi və rollara əsaslanan girişə nəzarətin (RBAC) tətbiqi araşdırılır. Ənənəvi sessiya əsaslı yanaşmalar server tərəfli yaddaş yüklənməsi, üfüqi miqyaslanma və mobil tətbiq inteqrasiyası baxımından əhəmiyyətli məhdudiyyətlər qoyur. Bu məhdudiyyətlərin aradan qaldırılması üçün JSON Web Token (JWT) standartı – RFC 7519 müasir RESTful veb arxitekturasında əsas autentifikasiya mexanizminə çevrilib. Məqalədə JWT-nin üç hissəli kriptoqrafik strukturu (başlıq, faydalı yük, imza), HMAC SHA-256 imzalama alqoritmi, token faydalı yükündə rol məlumatlarının kodlaşdırılması və əlavə şəbəkə dövriyyələri olmadan ön paneldə UI komponentinin yüklənməsini dinamik şəkildə idarə etmək üçün onun tətbiqi ətraflı təhlil edilir. Tədqiqat konkret təhlil nümunəsi kimi universitetin tədris iş yükünü təhlil etmək üçün hazırlanmış platformada müəllim-tələbə rol ayrılığının texniki tətbiqini təqdim edir və token yükündəki tək rol sahəsinin bütün interfeysin şaxələnmə qərarını necə idarə edə biləcəyini nümayiş etdirir.

Açar sözlər: JSON Web Token, autentifikasiya, rol-əsaslı giriş nəzarəti (RBAC), HMAC SHA-256, stateless arxitektura.

UOT: 004.056.5:004.738.5
JEL: L86, M15, O33
DOI: https://doi.org/10.54414/EPSQ8389

Giriş

İstifadəçi autentifikasiyası istənilən çox-istifadəçili veb-tətbiqin arxitektur mərkəzində yerləşir. Sistem "siz kimsiniz?" sualına cavab verdikdən dərhal sonra ikinci mühüm sual gəlir: "nə etməyinizə icazə verilir?" — bu, avtorizasiya problemidir. Hər ikisini birlikdə həll etmək bütün tətbiqin təhlükəsizlik sərhə-dini müəyyən edən identifikasiya-avtorizasiya arxitekturası yaradır [2, s.42].

Veb-tərtibat tarixinin böyük bir hissəsində sessiyaya əsaslanan autentifikasiya əsas yanaşma olub. Bu modeldə server hər bir uğurlu giriş üçün yaddaşda sessiya obyekti yaradır və müştəriyə, adətən, kukidə saxlanılan unikal sessiya identifikatoru göndərir. Müştəri bu identifikatoru hər sonrakı sorğuda təqdim edir və server onu müvafiq sessiya vəziyyətini axtarmaq üçün istifadə edir. Bu yanaşmanı tətbiq etmək asandır və çərçivələr tərəfindən yaxşı dəstəklənir, lakin onun üç əsas struktur məhdudiyyəti var.

Birincisi, server tərəfində yaddaş yükü: hər bir aktiv istifadəçi üçün sessiya obyekti saxlanılmalıdır, yəni yaddaş istehlakı eyni vaxtda istifadə edən istifadəçilərlə xətti miqyasda artır – bu, eyni vaxtda yüzlərlə tələbəyə xidmət göstərən universitet platfor-ması üçün ciddi narahatlıq doğurur. İkincisi, üfüqi miqyaslanma sürtünməsi: tətbiq yükləmə balanslayıcısının arxasındakı bir neçə serverdə işlədikdə, seansı yaradan serverdən başqa bir serverə yönləndirilən sorğu uyğun seans girişi tapmayacaq. Bunu həll etmək üçün ya yapışqan sessiyalar (hər bir müştərini daimi olaraq eyni serverə yönləndirmək), ya da Redis kimi xarici paylaşılan yaddaş tələb olunur – hər ikisi infrastrukturun mürəkkəbliyini artırır [3, s.88]. Üçüncüsü, mobil tətbiq empedansı: yerli mobil müştərilər kukiləri brauzerlərlə eyni yerli dəstək ilə idarə etmir, bu da sessiyaya əsaslanan axınların düzgün şəkildə tətbiq edilməsini çətinləşdirir [4, s.201].

JSON Veb Tokenləri tamamilə fərqli dizayn vasitəsilə hər üç məhdudiyyəti aradan qaldırır: token özündə bütün lazımi vəziyyəti daşıyır, odur ki, serverə əvvəlki qarşılıqlı əlaqələr haqqında heç bir yaddaş lazım deyil. Bu vətəndaşlığı olmayan xüsusiyyət JWT-ni müasir mikroservis və tək səhifəli tətbiq arxitekturalarında standart seçimə çevirir. Bu məqalədə JWT standartı dərindən təhlil edilir, onun rol kodlaşdırma qabiliyyəti araşdırılır və tək token sahəsinin müştəri ilə server arasında bütün avtorizasiya dövriyyələrini necə aradan qaldıra biləcəyi nümayiş etdirilir.

1. JWT-nin strukturu və kriptoqrafik əsasları. JSON Veb Tokeni nöqtələrlə ayrılmış üç Base64url kodlu seqmentdən ibarət yığcam, URL-etibarlı sətirdir: header.payload.signature. RFC 7519 standartı dəqiq müəyyən edir; hər bir seqmenti anlamaq həm təhlükəsizlik xüsusiyyətlərini, həm də formatın praktiki uyğunlaşmasını qiymətləndirmək üçün ilkin şərt hesab olunur [1].

Başlıq seqmenti tam olaraq iki sahəni ehtiva edir: typ, token növünü JWT kimi müəyyən edən və imzalama alqoritmini göstərən alg. Alqoritm seçiminin əhəmiyyətli təhlükəsizlik təsirləri var. HMAC SHA-256 (HS256) simmetrik bir alqoritmdir — eyni gizli açar həm imzalamaq, həm də yoxlamaq üçün istifadə olunur. Bu, yalnız bir serverin tokenləri yoxlaması lazım olduğu tək arxa sistem üçün uyğundur. RSA əsaslı asimmetrik alqoritmlər (RS256), əlavə açar idarəetmə mürəkkəbliyi bahasına yoxlama qabiliyyətini daha geniş şəkildə paylayır — açıq açarı saxlayan istənilən tərəf yoxlaya bilər. Tək API arxa hissəsinə malik universitet platforması üçün HS256 düzgün kompromis təmin edir [5, s.34].

Faydalı yük seqmenti iddiaları — istifadəçi və tokenin özü haqqında iddiaları ehtiva edir. RFC 7519 iddiaları üç qrupa bölür. Qeydiyyatdan keçmiş iddialar standartlaşdırılıb: iss (emitent), exp (bitmə müddəti), iat (verildiyi yer) və sub (mövzu) ən çox istifadə edilənlərdir. İctimai iddialar tərəflər arasında razılaşdırılır. Şəxsi iddialar tətbiqə xasdır; onlar tətbiqin standartın müəyyən etdiyindən kənara çıxan məlumatları daşıyırlar [1, s.9]. İki rollu təhsil platforması üçün faydalı yük aşağıdakı kimi qurulub: { "iss": "unilearn-api", "iat": 1716210000, "exp": 1716296400, "userId": 42, "role": "teacher" }. Rol sahəsi standart tərəfindən müəyyən edilməmiş, lakin ön tərəfin hansı interfeysi göstərəcəyinə qərar verməsi üçün lazım olan tək məlumatı daşıyan xüsusi bir iddiadır.

İmza seqmenti, gizli açardan istifadə edərək, seçilmiş alqoritmi kodlanmış başlıq və faydalı yükün birləşməsinə tətbiq etməklə yaradılır: HMACSHA256(base64UrlEncode(header) + '.' + base64UrlEncode(payload), secret). Əhəmiyyətli bir məqam: Base64url kodlaşdırması şifrələmə deyil. Tokenə sahib olan hər hansı bir tərəf faydalı yükü deşifrə edə və onun məzmununu oxuya bilər. İmza məxfiliyi qorumur - bütövlüyü qoruyur. İstifadəçi rol sahəsini 'tələbə'dən 'müəllim'ə dəyişdirərsə, yoxlama zamanı server tərəfindən hesablanan imza tokendəki ilə uyğun gəlməyəcək və sorğu rədd ediləcək [5, s.38].

2. Sessiya-əsaslı vs token-əsaslı: müqayisəli təhlil. Sessiya əsaslı və token əsaslı identifikasiya arasındakı memarlıq fərqi çox vaxt statuslu və statussuz kimi təsvir olunur. Sessiya əsaslı sistemdə server vəziyyəti saxlayır — hər bir aktiv sessiya serverin yaddaşında və ya ortaq yaddaşda bir girişdir. JWT sistemində token özü serverin ehtiyac duyduğu bütün vəziyyəti daşıyır; server sorğular arasında heç nə saxlamır [6, s.156].

Souza və Oliveira (2019) hər iki yanaşmadan istifadə edərək REST API tətbiqləri arasında müqayisəli benchmarking tədqiqatı aparıblar. Onların tapıntıları göstərib ki, JWT əsaslı identifikasiya ekvivalent yükləmə şəraitində server resurs istehlakını təxminən 40% azaldır, əsasən sessiya axtarış əməliyyatı - nə qədər sürətli olsa da - tamamilə aradan qaldırıldığı üçün [7, s.222]. Onlarla tələbənin eyni vaxtda viktorina təqdimatlarını təqdim edən platforma üçün bu fərq praktik olaraq əhəmiyyətlidir.

Ölçülənə bilənlik ölçüsündə JWT üstünlüyü həlledicidir. Token yoxlaması yalnız hər bir server nümunəsində olan gizli açarı tələb etdiyindən, yük balanslı klasterdəki istənilən qovşaq istənilən sorğunu müstəqil şəkildə yoxlaya bilər. Sinxronizasiya etmək üçün ortaq sessiya anbarı, konfiqurasiya etmək üçün yapışqan marşrutlaşdırma və saxlamaq üçün əlavə infrastruktur yoxdur. Klasterə yeni bir

server qovşağının əlavə edilməsi heç bir sessiya miqrasiyası tələb etmir [8, s.178].

JWT-nin təqdim etdiyi kompromis token ləğvidir. Sessiyanı mağazadan silməklə dərhal etibarsız hala gətirmək olar; istifadəçi hesabdan çıxsa belə, JWT, exp iddiası müddəti bitənə qədər qüvvədə qalır. Yüngülləşdirmə strategiyalarına qısa müddətli istifadə müddəti, token rotasiyasını yeniləmək və parol dəyişiklikləri kimi yüksək təhlükəsizlik əməliyyatları üçün kiçik bir ləğv siyahısının saxlanması daxildir - lakin bunlar sessiya rəhbərliyinin daha təbii şəkildə idarə etdiyi mürəkkəblik əlavə edir [9, s.67].

3. Rol məlumatının token payloadında kodlaşdırılması. JWT-nin çoxrollu sistemlər üçün təklif etdiyi ən memarlıq baxımından əhəmiyyətli imkan, rol məlumatlarını birbaşa token yükünə yerləşdirmək qabiliyyətidir. Bu, müştərinin identifikasiya etdiyi, sessiya identifikatoru aldığı və sonra identifikasiya olunmuş istifadəçinin hansı interfeysi göstərəcəyinə qərar verməzdən əvvəl hansı rolu tutduğunu öyrənmək üçün ayrıca GET /api/me sorğusu etməli olduğu sessiya əsaslı sistemlərdə geniş yayılmış nümunəni aradan qaldırır.

Burada araşdırılan təlimatçı-tələbə platformasında giriş nöqtəsi, yüklənməsində "müəllim" və ya "tələbə" olaraq təyin edilmiş rol sahəsi olan bir token qaytarır. Klient tərəfində işləmə sadədir: token localStorage-a yazılır; imzanı yoxlamadan yükü çıxarmaq üçün jwt_decode() çağırılır (imza yoxlaması yalnız serverin məsuliyyətidir); rol sahəsi oxunur; və window.location.href ya /teacher/dashboard, ya da /student/dashboard olaraq təyin edilir. Bütün marşrutlaşdırma qərarı əlavə şəbəkə sorğusu olmadan tək bir funksiya çağırışında tamamlanır [10,s.89].

Bu ayrılmanın ikinci dərəcəli üstünlüyü var: ön və arxa hissə müstəqil şəkildə inkişaf edə bilər. Gələcəkdə üçüncü bir rol - məsələn, şöbə administratoru - əlavə olunarsa, arxa hissə 'admin' rolu olan tokenlər göndərəcək və ön hissənin yönləndirmə məntiqi yalnız bir əlavə şərti budaq tələb edəcək. Doğrulama son nöqtəsində, token strukturunda və ya doğrulama ara proqramında heç bir dəyişiklik tələb olunmayacaq.

4. Frontend-də rol-əsaslı UI yüklənməsi. Klient tərəfindəki rol marşrutlaşdırması təhlükəsizlik nəzarəti deyil, rahatlıqdır. /teacher/dashboard URL-ni bilən istifadəçi, giriş axınının nəyi nəzərdə tutduğundan asılı olmayaraq, onu birbaşa yaza bilər. Hər qorunan son nöqtəyə edilən hər bir sorğuda serverdə həqiqi rol tətbiqi baş verməlidir [11, s.124]. authMiddleware funksiyası tokeni Authorization: Bearer <token> başlığından çıxarır, gizli açarla jwt.verify() funksiyasını çağırır və doğrulama uğurlu olarsa, deşifrə edilmiş yükü req.user-ə əlavə edir. jwt.verify() eyni anda iki yoxlama aparır: imzanı təsdiqləyir (tokenin dəyişdirilmədiyini təmin edir) və exp iddiasını yoxlayır (tokenin müddəti bitmədiyini təmin edir). Hər iki uğursuzluq 403 Qadağan olunmuş cavabla nəticələnir [12, s.201].

İkinci ara proqram təminatı təbəqəsi — roleMiddleware — aşağı axında yerləşir və req.user.role-nu müəyyən marşrut qrupu tərəfindən tələb olunan rola qarşı yoxlayır. Təlimatçı marşrutları rol === 'müəllim' tələb edir; tələbə tokenindən gələn hər hansı bir sorğu marşrut işləyicisi işə salmazdan əvvəl 403 ilə rədd edilir. Bu ikiqatlı yanaşma, təhlükəsizliyin hər bir problem üçün kod bazasında tam bir yerdə tətbiq olunduğu deməkdir: authMiddleware token etibarlılığına, roleMiddleware icazə yoxlamasına sahibdir. Nə marşrut işləyiciləri, nə də ön tərəf bu məntiqi təkrarlamağa ehtiyac duymur.

Klient tərəfindəki authGuard funksiyası server tətbiqinin əvəzinə deyil, üstündə istifadəçi təcrübəsi təbəqəsi təmin edir. Tokeni localStorage-dan oxuyur, onu deşifrə edir, exp sahəsini Date.now() ilə yoxlayır və rol sahəsini cari səhifə üçün gözlənilən dəyərə qarşı yoxlayır. Uyğunsuzluq dərhal giriş səhifəsinə yönləndirir. Bu, tələbənin təlimatçı URL-inə keçməsi, hər məlumat yüklənməsi zamanı API-dən 403 alması və boş və ya səhvlə dolu səhifə görməsi kimi çaşdırıcı təcrübənin qarşısını alır - lakin faktiki məlumatlara girişi dayandıran serverdir.

5. Təhlükəsizlik məsələləri. JWT-nin statussuz dizaynı sessiya əsaslı sistemlərin fərqli şəkildə idarə etdiyi təhlükəsizlik mülahizələri sinfini təqdim edir. Düzgün tətbiq üçün onları anlamaq vacibdir. XSS və

localStorage məruz qalması. Tokenlərin localStorage-da saxlanması onları səhifədə işləyən istənilən JavaScript üçün əlçatan edir. Uğurlu bir XSS inyeksiyası - tətbiqin mənşəyində hücumçu tərəfindən idarə olunan skriptin icrası - tokeni oxuya və çıxara bilər. Standart azaldılma, server tərəfindəki giriş dezinfeksiyası ilə birləşdirilmiş ciddi Məzmun Təhlükəsizlik Siyasəti başlıqlarıdır ki, hücumçu tərəfindən idarə olunan skript əvvəlcə icra edə bilməsin [13, s.56]. HttpOnly kukiləri localStorage məruz qalmasının qarşısını alır, lakin CSRF riskini yenidən təqdim edir; düzgün seçim tətbiqin təhdid modelindən asılıdır.

Yük manipulyasiyası. Bölmə 1-də qeyd edildiyi kimi, Base64url kodlaşdırması açar olmadan geri qaytarıla bilər. İstifadəçi faydalı yükü deşifrə edə, rol sahəsini dəyişdirə, yenidən kodlaya və yeni bir token qura bilər. Lakin imza gizli açar olmadan yenidən hesablana bilməz, buna görə də jwt.verify() dəyişdirilmiş tokeni rədd edəcək. Bu qorunma yalnız gizli açar kifayət qədər güclü olduqda və heç vaxt açıqlanmadığı müddətcə qüvvədədir. Açar kriptoqrafik cəhətdən təhlükəsiz təsadüfi ədəd generatoru ilə yaradılmalı, ən azı 256 bit uzunluğunda olmalı, yalnız mühit dəyişənlərində saxlanılmalı və heç vaxt mənbə depolarına yerləşdirilməməlidir [5, s.42].

Tokenin müddəti və ləğvi. Müddət tələbi istifadəçi rahatlığı ilə ifşa pəncərəsini balanslaşdıran bir dəyərə təyin edilməlidir. Burada araşdırılan platformada istifadə edildiyi kimi, 24 saatlıq müddət oğurlanmış tokenin bir günə qədər etibarlı olması deməkdir. Daha qısa müddətli müddətlər (15 dəqiqə) və yeniləmə tokeni mexanizmi bu pəncərəni daha tez-tez token mübadiləsi hesabına azaldır. Universitet qiymətləndirmə platforması üçün 24 saat məqbul bir güzəşt hesab edildi; bank və ya tibbi tətbiq əhəmiyyətli dərəcədə daha qısa müddət tələb edərdi [14, s.312].

Nəticə

JWT token əsaslı identifikasiya müasir veb tətbiqinin hazırlanması üçün standart həllə çevrilib və burada təqdim olunan təhlil bunun səbəbini izah edir. Onun statussuz arxitekturası server tərəfindəki sessiya yaddaşını aradan qaldırır, miqyaslanma xüsusiyyətləri düyünlər arasında ortaq bir infrastruktur tələb etmir və faydalı yükü əlavə şəbəkə sorğuları olmadan aşağı axın qərarlarını idarə edən tətbiqə xas məlumatları - istifadəçi rolu da daxil olmaqla - daşıyır.

Təlimatçı-tələbə platforması üzrə iş araşdırması göstərir ki, rol əsaslı interfeys ayrılması token faydalı yükündə tək bir özəl iddia vasitəsilə təmiz şəkildə həyata keçirilə bilər, marşrutlaşdırma məntiqi bir neçə klient tərəfindəki JavaScript sətri ilə məhdudlaşır və serverdəki iki fokuslanmış ara proqram funksiyası tərəfindən idarə olunan faktiki təhlükəsizlik təminatı. Narahatlıqların bu ayrılması - klient üçün rahatlıq, serverdəki təminat - tətbiq böyüdükcə düzgün şəkildə miqyaslanan modeldir.

Gələcək işlər istifadəçi təcrübəsini pisləşdirmədən məruz qalma pəncərələrini azaltmaq üçün yeniləmə token fırlanma strategiyalarını, çıxış və sessiyanın etibarsızlaşdırılması ssenariləri üçün token ləğv siyahısı dizaynlarını və çoxfaktorlu identifikasiya axınlarının JWT əsaslı sistemlərlə inteqrasiyasını araşdırmalıdır - statussuz modelin statuslu sessiyaların daha təbii şəkildə təmin etdiyinə dair zəmanətlərə nail olmaq üçün qəsdən əlavə mexanizmlər tələb etdiyi sahələr.

ƏDƏBİYYAT SİYAHISI

1. Bradley J., Sakimura N., Jones M. JSON Web Token (JWT) — RFC 7519. IETF. 2015. https://datatracker.ietf.org/doc/html/rfc7519
2. Ferraiolo D., Kuhn R., Chandramouli R. Role-Based Access Control (2nd ed.). Artech House. 2007.
3. Tanenbaum A.S., Van Steen M. Distributed Systems: Principles and Paradigms (3rd ed.). Pearson. 2016.
4. Meier J.D., Homer A., Hill D., Taylor J., Bansode P., Wall L., Boucher Jr.R., Bogawat A. Web Application Architecture Guide. Microsoft Patterns & Practices. 2011. Microsoft Application Architecture Guide (2nd ed.). Microsoft Press. 528 p.
5. Jones M., Bradley J., Sakimura N. JSON Web Signature (JWS) — RFC 7515. IETF. 2015.

https://datatracker.ietf.org/doc/html/rfc7515
6. Richardson L., Ruby S. RESTful Web Services. O'Reilly Media. 2007.
7. Souza F., Oliveira R. Performance comparison of session-based and token-based authentication in RESTful APIs. Journal of Web Engineering, 2019;18(3):215–238.
8. Newman S. Building Microservices (2nd ed.). O'Reilly Media. 2021
9. Auth0. Refresh Token Rotation. 2023. https://auth0.com/docs/secure/tokens/refresh-tokens/refresh-token-rotation
10. Flanagan D. JavaScript: The Definitive Guide (7th ed.). O'Reilly Media. 2020.
11. Wieruch R. The Road to React. Self-published. 2020.
12. Express.js Team. Express 4.x API Reference. 2023. https://expressjs.com/en/4x/api.html
13. Stuttard D., Pinto M. The Web Application Hacker's Handbook (2nd ed.). Wiley. 2011.
14. Hardt D. (Ed.). The OAuth 2.0 Authorization Framework — RFC 6749. IETF. 2012. https://datatracker.ietf.org/doc/html/rfc6749

Bakhish Tahir JAFAROV
Master's student at Western Caspian University

TOKEN-BASED AUTHENTICATION AND ROLE-BASED ACCESS CONTROL IN WEB APPLICATIONS: ANALYSIS AND IMPLEMENTATION OF THE JWT STANDARD

Summary

This study examines the fundamental problem of user identity verification in modern multi-user web applications — the selection of an authentication mechanism and the implementation of role-based access control (RBAC). Traditional session-based approaches impose significant constraints in terms of server-side memory overhead, horizontal scalability, and mobile application integration. To overcome these limitations, the JSON Web Token (JWT) standard — RFC 7519 — has become the dominant authentication mechanism in modern web architecture. The paper provides a detailed analysis of JWT's three-part cryptographic structure (header, payload, signature), the HMAC SHA-256 signing algorithm, the encoding of role information in the token payload, and its use for dynamically controlling UI component loading on the frontend. The research presents the technical implementation of the instructor-student role separation in a platform developed for analysing a university's teaching workload as a concrete case study.

Keywords: JSON Web Token, authentication, role-based access control (RBAC), HMAC SHA-256, stateless architecture.

Бахыш Тахир ДЖАФАРОВ
Магистрант Западно-Каспийского Университета

АУТЕНТИФИКАЦИЯ НА ОСНОВЕ ТОКЕНОВ И РОЛЕВОЙ КОНТРОЛЬ ДОСТУПА В ВЕБ-ПРИЛОЖЕНИЯХ: АНАЛИЗ И ПРИМЕНЕНИЕ СТАНДАРТА JWT

Резюме

В данной работе исследуется фундаментальная проблема аутентификации пользователей в современных многопользовательских веб-приложениях — выбор механизма аутентификации и реализация ролевого контроля доступа (RBAC). Традиционные подходы на основе сессий создают значительные ограничения с точки зрения нагрузки на серверную

память, горизонтального масштабирования и интеграции с мобильными приложениями. Для преодоления этих недостатков стандарт JSON Web Token (JWT) — RFC 7519 — стал доминирующим механизмом аутентификации в современной веб-архитектуре. В статье подробно анализируется трёхчастная криптографическая структура JWT (заголовок, полезная нагрузка, подпись), алгоритм подписи HMAC SHA-256, кодирование ролевой информации в payload токена и её использование для динамического управления загрузкой компонентов пользовательского интерфейса. Исследование представляет реализацию разделения ролей преподаватель-студент в платформе для анализа учебной нагрузки университета в качестве конкретного примера.

Ключевые слова: JSON Web Token, аутентификация, ролевой контроль доступа (RBAC), HMAC SHA-256, stateless архитектура.

Daxil olub: 07.04.2026