

Report

Penetration test Reducate application

17-04-2025 VERSION 1.0

Versioning

Version	Date	Status	Author	Description
0.1	08-04-2025	Draft	Kevin van Straten	Initial version
0.9	15-04-2025	Draft	Bas Cijssouw	QA version
1.0	17-04-2025	Final	Kevin van Straten	Definitive version

The S-Unit internal documents use minor version numbers (e.g. 0.1). After internal quality assurance has successfully taken place, the major version number is incremented (e.g. 1.0) and reaches the status 'final'.

Distribution

Version	Date	Recipients
1.0	17-04-2025	Stephan Csorba



This penetration test is conducted according to the established quality standards of the CCV certification. The CCV certification guarantees the quality of the test execution, and explicitly does not offer any statements or guarantees regarding the quality or security of the tested object itself.

Contents

1. Management Summary	4
1.1. Results	4
1.2. Conclusion and Advice	5
2. Introduction and Scope	6
2.1. Scope deviation	7
3. Approach	8
4. Findings	9
4.1. Reducate application	9
4.1.1. Mass assignment in edit profile form	9
4.1.2. Stored XSS on events and certificates	12
4.1.3. Course subscription validation bypass	14
4.1.4. Insecure logout mechanism	16
4.1.5. Missing security headers	18
5. Conclusion	19
5.1. Research questions	19
5.1.0. Overall answer to the research questions	19
5.1.1. Access to sensitive data from other students as a student	19
5.1.2. Start or follow a course after subscription has expired	19
5.1.3. Access to unassigned course materials	19
5.1.4. Manipulating data of students as a manager	20
5.1.5. Access to information about students of other organizations	20
5.2. Remaining observations	20
5.2.1. Long expiration JWT token LRS	20
5.2.2. Bypassing course process	20
5.2.3. Discrepancy between Student UI and other UIs	20
5.3. Open questions	20
5.4. Recommendations	21
APPENDIX A. Risk Classification of Vulnerabilities	22
APPENDIX B. Impact	23

1. Management Summary

The S-Unit has performed a penetration test for Reducate Services B.V. on the Reducate web application. The purpose of the test was to gain insight into vulnerabilities mainly related to authorization and organization separation. The test was performed between 08-04-2025 and 14-04-2025 with a total time investment of 6 test days.

1.1. Results

The following table summarizes the findings of the penetration test. See **Appendix A** for more information on the risk column.

Ref.	Finding	Risk	Summary
4.1.1	Mass assignment in edit profile form	Critical	The Reducate application is vulnerable to complete takeover of a tenant due to improper validation when editing a profile. This finding was communicated and resolved during the pentest.
4.1.2	Stored XSS on events and certificates	Medium	The events and certificates pages within the Student UI insecurely handle user input and one of those pages can be used by a manager to attack other students of the application. A manager can use this to alter details of students or their courses.
4.1.3	Course subscription validation bypass	Medium	The Reducate application fails to properly validate a student's subscription, allowing students to access any course available within the tenant. The risk is reduced because the student needs knowledge of the course ID and lesson ID, which are semi-random values and are therefore not easily enumerated.
4.1.4	Insecure logout mechanism	Low	The Reducate application does not properly invalidate or delete a user's sessions upon logout. If an attacker manages to obtain a valid session cookie, for example in a shared browser environment, this can be used to keep access to the application even after the user has logged out.
4.1.5	Missing security headers	Low	The Reducate application exposes its users to security risks due to missing security headers. This was used to successfully exploit finding 4.1.2 and can also be used to intercept network traffic to and from the application in very specific scenarios.

1.2. Conclusion and Advice

During the penetration test, a critical vulnerability was identified that allowed any user with the ability to edit their profile to gain tenant-level access and therefore cause impact across all research questions (finding 4.1.1). This finding was communicated to Reducate during the pentest and has since been resolved.

Several other vulnerabilities have been identified during the pentest, allowing a student to access courses that are not assigned to the student or without a valid subscription (finding 4.1.3) and allowing managers to gain access to the application in the context of other students (finding 4.1.2).

Except for the critical finding, no methods were found for students to gain access to sensitive data of other students and for managers to gain access to information regarding (students of) other organizations. Several other findings were done that did not cause direct impact on any of the research questions, including not properly invalidating sessions during logout (finding 4.1.4) and unnecessarily exposing users to security risks due to missing security headers (finding 4.1.5).

It is worth noting that multiple remaining observations were made during this pentest. First, the LRS service uses a token with a long lifetime, increasing the impact when an attacker gains access to such a token. Second, course validation was found to be implemented heavily on the client side, allowing students to manipulate course progress or exam answers. Even though Reducate has accepted this as a known risk, this can still cause impact for other organizations using the application and relying on course integrity. Last, there was a clear discrepancy between the Student UI and other UIs, such that several vulnerabilities found within the Student UI were properly mitigated within other UIs. This might be an indication that security within the Student UI is not prioritized to the same extent as other UIs.

Based on the above, The S-Unit advises the following steps:

1. Apply the recommended changes for the individual findings, prioritizing those with the highest impact to efficiently minimize the risk
2. Investigate to which extent the lifetime of JWT tokens should be 1 month or whether this can be reduced to a shorter timeframe
3. Consider implementing verification of course progress and exam answers on the server side, thereby preventing students from manipulating courses
4. Investigate if and why security within the Student UI is notably different compared to other UIs and consider implementing the same, strict security checks across all UIs

2. Introduction and Scope

Reducate Services B.V. has requested The S-Unit to perform a penetration test on the Reducate application, consisting of a Laravel platform using Livewire¹, a React player, a Laravel LRS API and FusionAuth SSO implementation. The penetration test was performed between 08-04-2025 and 14-04-2025 with a total time investment of 6 test days.

Research questions for the penetration test were:

- To which extent can a student gain unauthorized access to sensitive data of other employees, such as certificates or course materials?
- To which extent can a student start or follow a course after the subscription of that student has expired?
- To which extent can a student or manager gain unauthorized access to course material which is not explicitly assigned to these users?
- To which extent can a manager manipulate data, such as altering or deleting students or courses?
- To which extent can a manager gain unauthorized access to information about (students of) other organizations?

The goal of the penetration test was to:

1. Gain insight in vulnerabilities and risks for the inspected systems.
2. Get advice about how to improve security for the systems in question. In other words: how can the vulnerabilities and risks found during testing best be mitigated

The scope of the penetration test was limited to the following components:

- Reducate application with different brands:
 - Puritas Student UI at <https://student.puritas-college-acc.com/>
 - Puritas Manager UI at <https://manager.puritas-college-acc.com/>
 - E-wise Student UI at <https://student.ewise-acc.com/>
 - E-Wise Manager UI at <https://manager.ewise-acc.com/>
- LRS API at <https://lrs.ewise-acc.com/>
- SSO implementation at <https://reducate.fusionauth.io/>

The penetration test was conducted from the following user roles:

- Unauthenticated user
- Authenticated student user
- Authenticated manager user

The following was explicitly excluded from the scope of the penetration test:

- Social engineering / phishing attacks
- Denial of Service

¹ <https://laravel-livewire.com/>

The penetration test was conducted from the following IP addresses:

- 217.67.243.148 (External office)
- 20.86.84.109 (Nessus scanner)
- 141.138.141.211 and/or 2a01:7c8:aaaa:34d:5054:ff:fe46:ead7 (VPN gateway)
- 20.82.51.44/30 (External callbacks)
- 149.210.173.106 and/or 2a01:7c8:aab4::10b/64 (DNS)
- 149.210.173.241 (NAT-VPN A)
- 149.210.173.247 (External-VPN A)
- 149.210.173.6 (External-VPN B)
- 149.210.173.3 (Burp Suite Collaborator)

For the penetration test, Reducate Services B.V. made the following changes from the default environment:

- Whitelisting the IP addresses of The S-Unit

2.1. Scope deviation

While interacting with the player during the penetration test, several other APIs were found to be used by the application. Although these APIs were not explicitly included within the scope of this pentest, they did provide nearly all functionality regarding course progression and verification. Therefore, the following URLs were also placed in scope for the penetration test:

- <https://api.ewise-acc.com/>
- <https://editor-api.ewise-acc.com/>

3. Approach

The following main steps were taken during the penetration test:

- Intake and acceptance scope
- Penetration test on Reducate application
- Report writing
- Quality assessment report
- Adjustments report and delivery

4. Findings

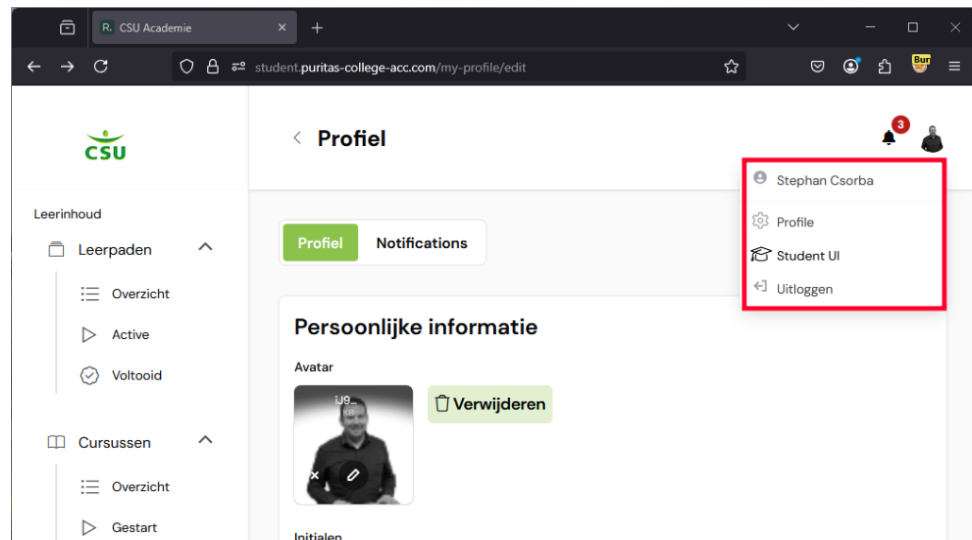
4.1. Reducate application

4.1.1. Mass assignment in edit profile form

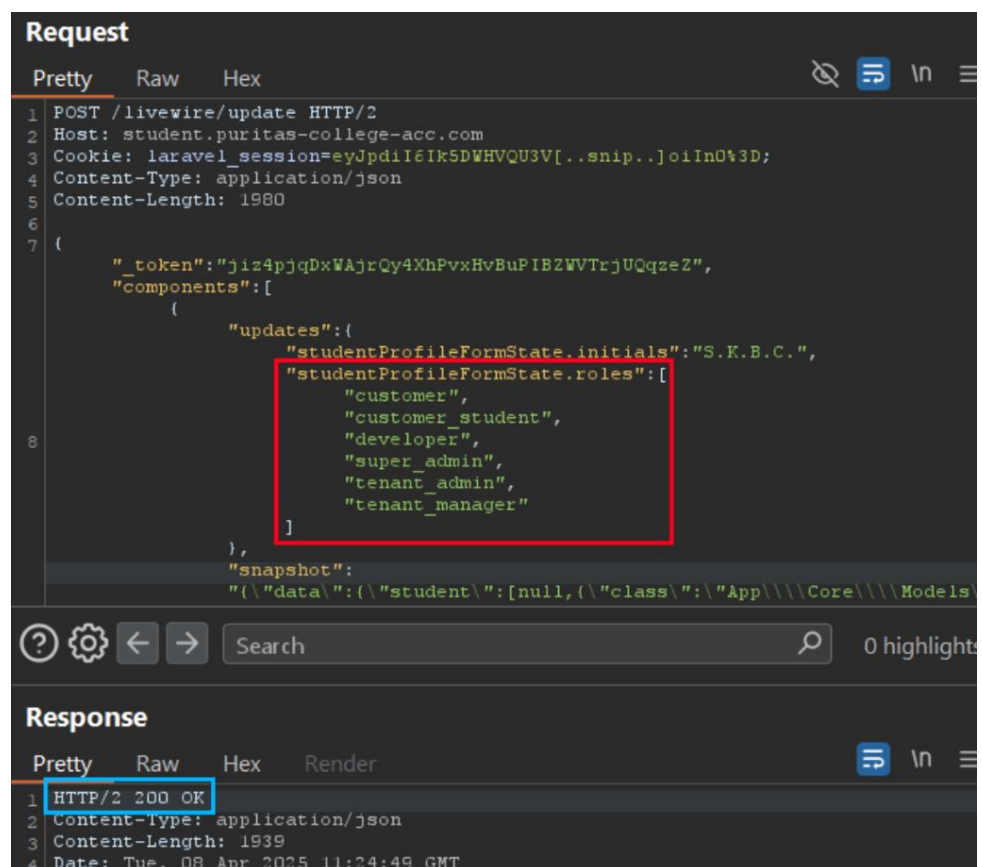
Classification	Critical
CVSS Score	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H
Impact	Security Bypass, Exposure of Sensitive Information
Summary	<p>The Reducate application is vulnerable to complete takeover of a tenant due to a mass assignment vulnerability in the edit profile functionality. The Reducate application allows users to change their profile within the Student UI at <code>/my-profile/edit</code>. Analysis of this page uncovered that the associated save function of the Livewire component <code>edit-student-profile</code> saves all properties from the Livewire <code>updates</code> object to the database, without checking whether the specified properties are actually supposed to be modified by the form on the page.</p> <p>This means that arbitrary attributes of the associated user can be modified, including the <code>roles</code> attribute in which the user's permissions are stored.</p> <p>During the penetration test, this was successfully exploited to allow a student user to escalate its privileges to tenant admin rights.</p> <p>This finding was communicated to Reducate during the penetration test and has since been resolved.</p>
Risk	An authenticated attacker capable of modifying their profile through the Student UI can use this finding to escalate to tenant admin privileges and completely take over the tenant. In the context of the application this means the ability to manage all users, associated brands and customers. In the context of the developer permissions this also includes access to FusionAuth information, server config, PHP server variables and environment variables.
Advice	Ensure that intended security restrictions cannot be bypassed. In the context of Livewire forms this means that forms with intended write restrictions should validate these restrictions in their corresponding save call. Consider using a whitelist approach in combination with the <code>only²</code> method.
Screenshot	The screenshots on the following pages show a student user elevating their rights to tenant admin and accessing the Tenant UI by abusing the mass assignment vulnerability.

² <https://livewire.laravel.com/docs/forms#adding-validation>

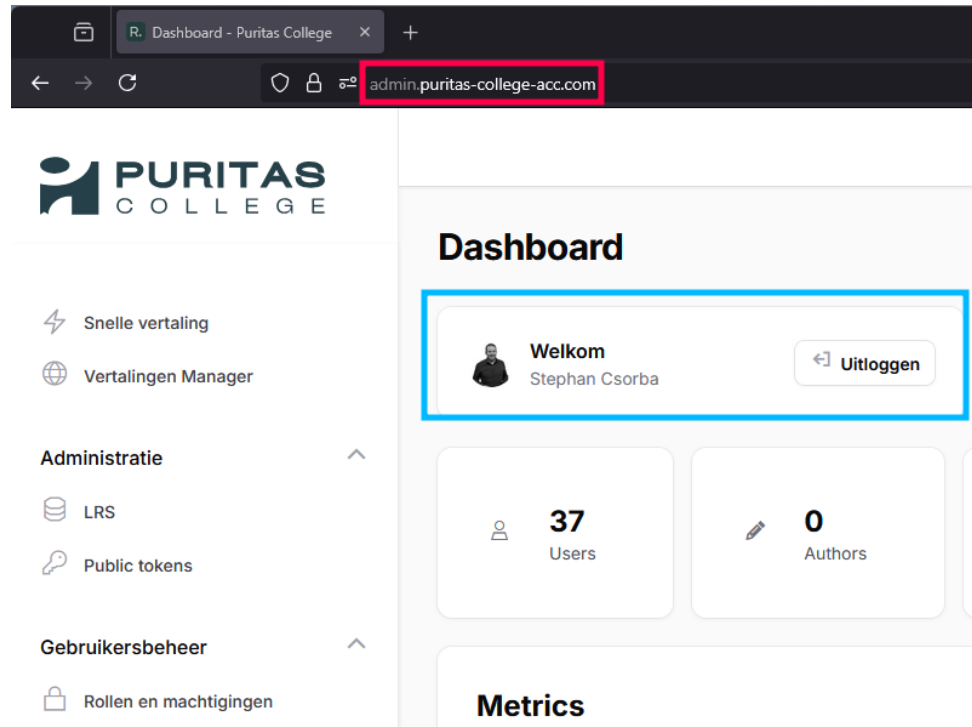
The first screenshot shows the profile page of the student user. The area outlined in red shows that the user only has access to the Student UI.



The second screenshot shows a request to change the student user's profile. The area outlined in red shows the additional property that is added as part of the payload. The area outlined in blue shows that the server accepted these changes.



The last screenshot shows the Tenant UI of the application. The area outlined in red shows the URL of the Tenant UI, which can now be accessed by the student user. The area outlined in blue shows the same user profile as the first screenshot. Note that the student user now has tenant admin privileges.

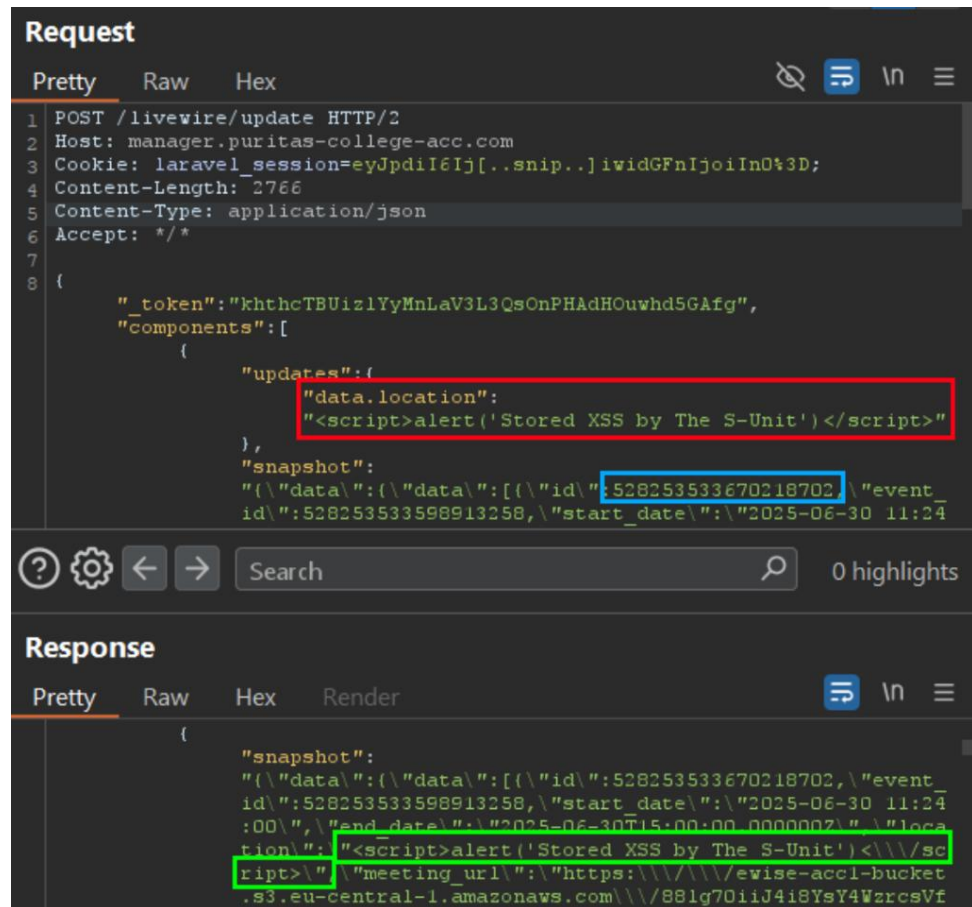


4.1.2. Stored XSS on events and certificates

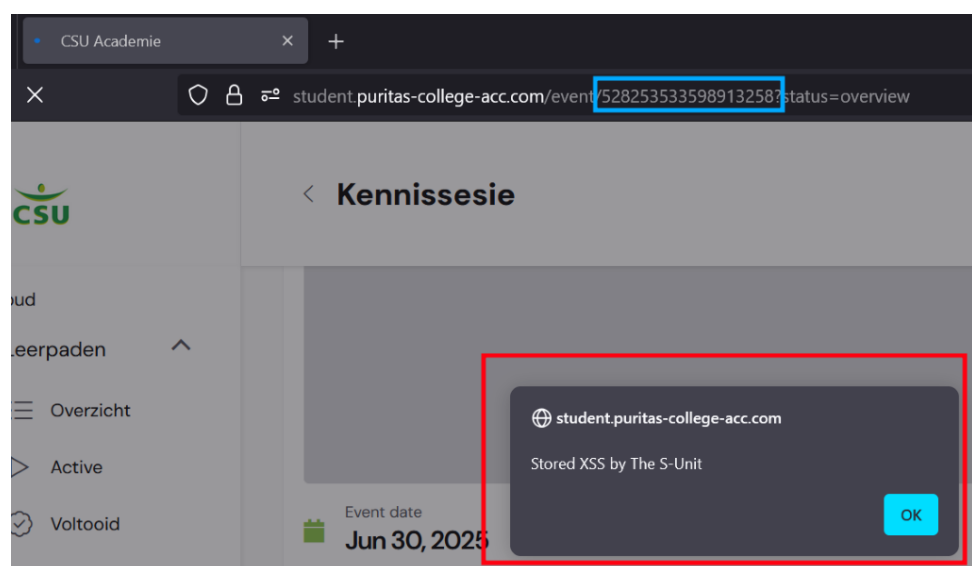
Classification	Medium
CVSS Score	CVSS:3.1/AV:N/AC:L/PR:H/UI:R/S:U/C:L/I:H/A:N
Impact	Cross-Site Scripting
Summary	<p>Several pages within the Student UI of the Reducate application are vulnerable to Stored Cross-Site Scripting (XSS). The Reducate application allows managers to create events to which students can register. The application also allows students to upload their own certificates for external courses that they finished. During analysis of both functionalities, the following was uncovered:</p> <ul style="list-style-type: none">- The functionalities of the events pages at <code>/events</code> and <code>/events/<eventid></code> directly add the <code>data.location</code> value of an event to the contents of an <code>h4</code> tag without properly encoding this for the context of an HTML tag.- The functionality of the certificates page at <code>/certificates</code> directly adds the <code>certificateData.course_provider</code> value of a certificate to the contents of a <code>span</code> tag without properly encoding this for the context of an HTML tag. <p>This means that the <code>data.location</code> and <code>certificateData.course_provider</code> fields can be used to inject custom HTML and JavaScript into the corresponding pages.</p>
Risk	<p>An authenticated attacker with the manager role can use this finding to gain access to the data and functionality of all users that access the mentioned <code>/events</code> page. In the context of events, a manager can use this to attack other students within the organization. In the context of certificates, this cannot be used by students to attack other users as certificates can only be viewed by the user that uploaded it.</p> <p>The classification is increased due to the possibility of managers to alter student information or their courses through this finding and the direct correlation to the corresponding research question.</p> <p>The classification is lowered because the session cookies are properly protected with the <code>HttpOnly</code> and <code>Secure</code> flags, therefore only allowing access as long as the targeted user remains on the vulnerable page.</p>
Advice	<p>Ensure that user input cannot have any special meaning within the context it is used in. In the context of HTML tag content this means that at least the tag delimiters (<code><</code> and <code>></code>) should be replaced with their corresponding HTML entities.</p>

Screenshot

The following screenshot shows the creation of an event. The area outlined in red shows the payload injected into the location field. The area outlined in blue shows the event id. The area outlined in green shows that the payload was successfully submitted to the application.



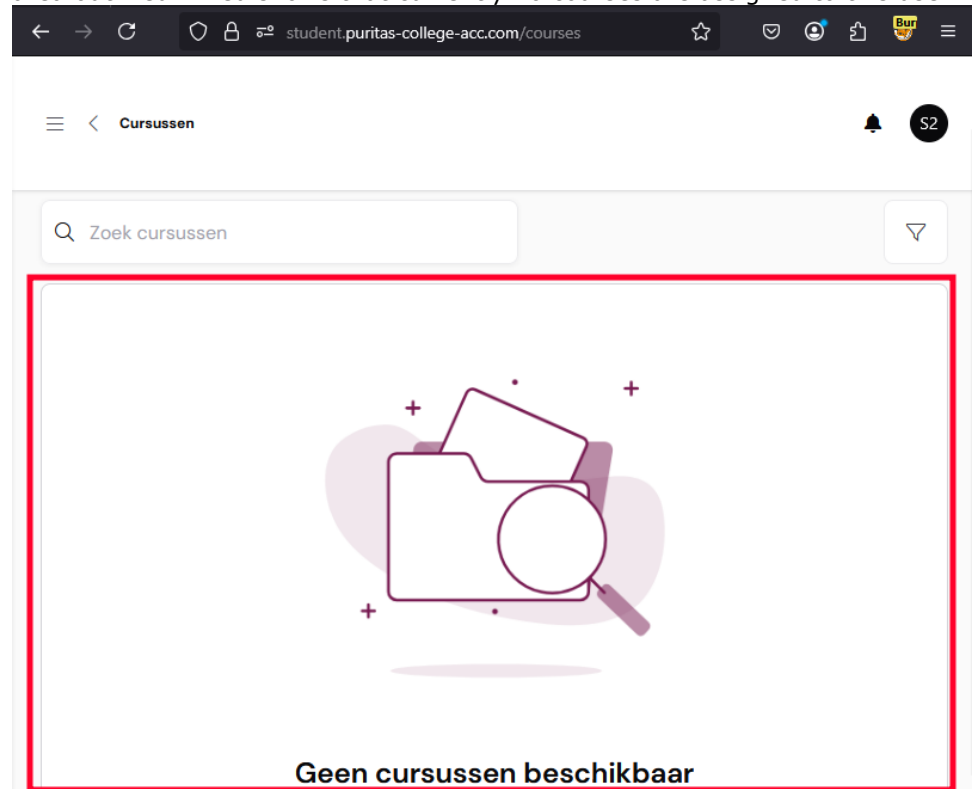
The following screenshot shows an overview of the specific event that was created in the first screenshot. The area outlined in red shows the XSS pop-up. The area outlined in blue shows the same event id as in the first screenshot.



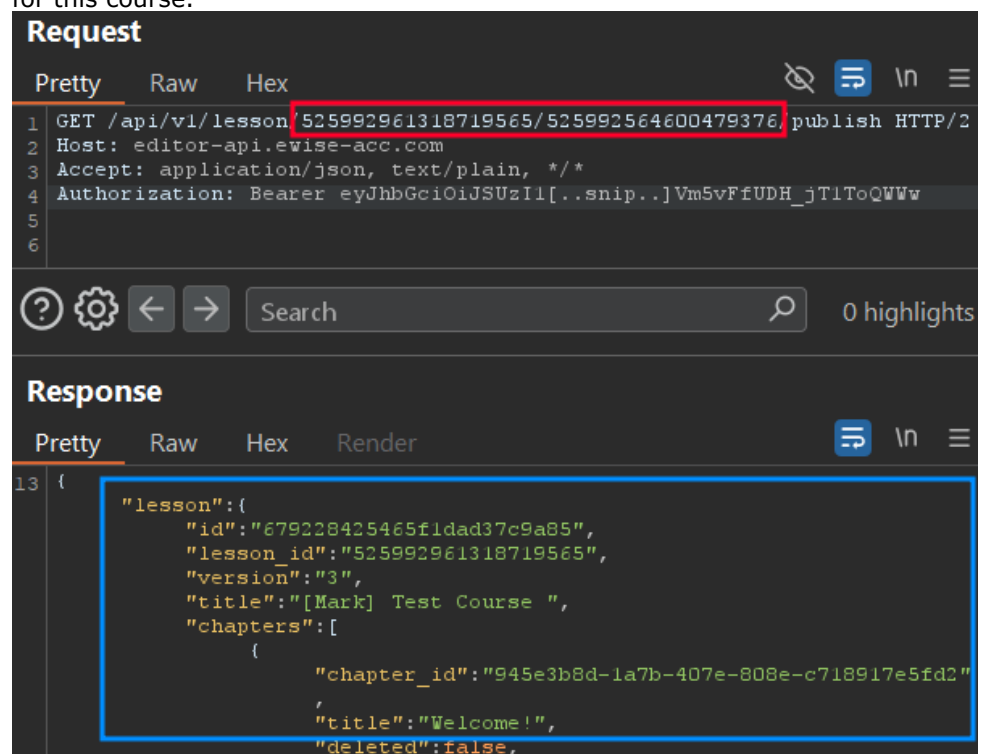
4.1.3. Course subscription validation bypass

Classification	Medium
CVSS Score	CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:N/A:N
Impact	Security Bypass, Exposure of Sensitive Information
Summary	<p>The Reducate application fails to properly validate a student's subscription, allowing students to access all courses available to the player and LRS API. The Reducate application allows students to start or follow courses for which they have a valid subscription. The retrieval of a course consists of the following relevant steps taken by the application:</p> <ul style="list-style-type: none">- The application sends a request to the Livewire component <code>course-control-button</code>, which returns a URL to the player that contains the associated course ID and lesson ID to the course.- When loading the mentioned URL, the application sends a request to the <code>api.ewise-acc.com/v1/lesson/<lessonID>/validate</code> endpoint. The API can either respond with a 200 OK or 403 Forbidden message based on if the user has a valid subscription.- If the response is 200 OK, further requests are sent to <code>api.ewise-acc.com</code> and <code>lrs.ewise-acc.com</code> to load the actual course content. <p>Analysis of this process uncovered the following two mistakes:</p> <ul style="list-style-type: none">- The verification process of a student's subscription solely depends on the response of the eWise API call to the <code>/validate</code> endpoint, which can be manipulated. This allows for the retrieval of any course through the player.- Subsequently, the LRS and eWise APIs perform no additional verification regarding the validity of the student's subscription when loading course content. This allows for direct retrieval of course information through these APIs. <p>This means that a student can load all course content from the player, given the knowledge of a valid course ID and lesson ID.</p>
Risk	<p>An authenticated attacker, with knowledge of a course ID and lesson ID, can use this finding to bypass the subscription validation and access the respective course materials.</p> <p>The classification is increased due to direct relevance to the research questions regarding unauthorized course access.</p> <p>The classification is decreased because the course ID and lesson ID are semi-random values and therefore cannot be easily enumerated.</p>
Advice	<p>Ensure that intended security measures cannot be bypassed. This means verification of access to restricted content should be implemented server side and by the respective services that deliver said content. In the context of course material this means that verification must be done by the respective APIs. Consider implementing server-side verification regarding the player and the verification process.</p>
Screenshot	<p>The screenshots on the following page show that even though a student is currently not assigned to any courses, they can still be accessed directly through the API.</p>

The following screenshot shows the current courses for a student user. The area outlined in red shows that currently no courses are assigned to this user.



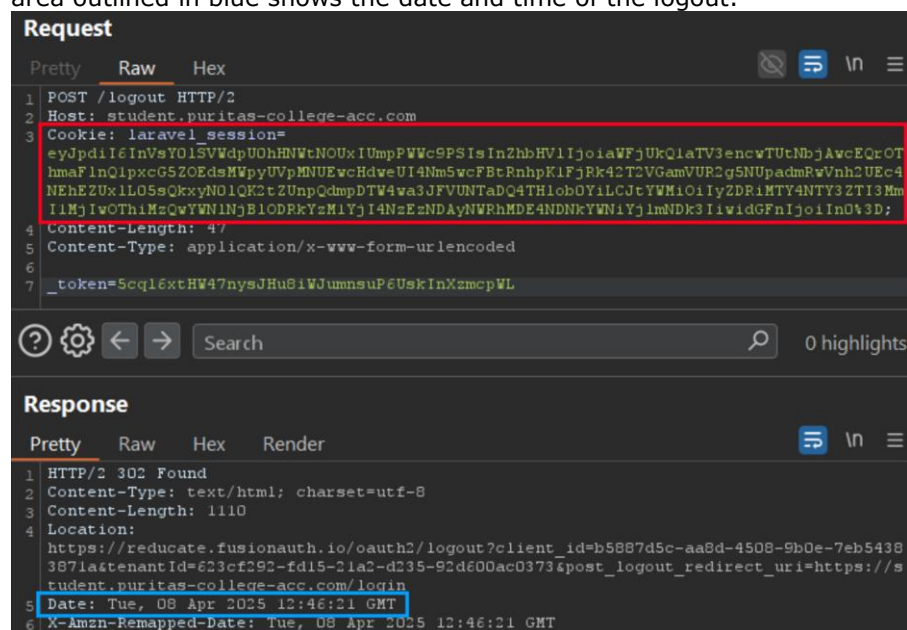
The following screenshot shows a request to the eWise API to get course content. The area outlined in red shows the specific course ID and lesson ID for this course. The area outlined in blue shows that the API returns the course content, regardless of the user not having a valid subscription or assignment for this course.



4.1.4. Insecure logout mechanism

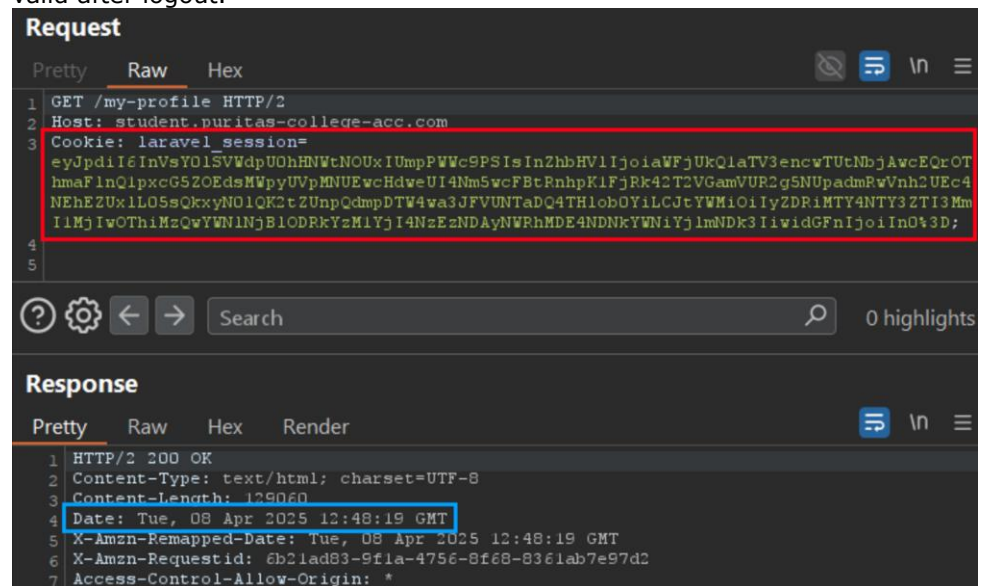
Classification	Low
CVSS Score	CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:L/A:N
Impact	Security Bypass, Hijacking
Summary	<p>The Reducate application does not properly invalidate or delete session cookies and tokens upon logout, resulting in an increased risk of session reuse or hijacking. The Reducate application uses FusionAuth SSO with the following session identifiers for authentication and authorization within the application:</p> <ul style="list-style-type: none">- A <code>laravel_session</code> cookie for use within the platform itself, with an expiration time of 7 days- A JWT-token for use within the LRS API, with an expiration time of 1 month <p>During analysis of the logout functionality, it was found that while the <code>laravel_session</code> cookie is overwritten on the client-side, it does not get deleted or invalidated on the server side. Furthermore, it was found that the JWT-token of the LRS API does not get invalidated on the server side. This means that sessions stay valid after a logout until their specified expiry date.</p>
Risk	An attacker, that manages to obtain a valid session cookie or token of a user, can use this finding to keep access to the application even after a user has logged out. This attack is mainly relevant in situations such as shared browser environments.
Advice	Ensure that intended security measures cannot be bypassed. In the context of logout functionality in combination with SSO, this means that all relevant sessions should be deleted both on the application and the SSO provider. For session cookies, this means deleting the cookies server side. For JWT-tokens, this can be done by using a blacklist of invalid tokens based on the <code>jti</code> ³ claim.
Screenshot	The following screenshots show that the session cookie remains valid even after a user has successfully logged out.

The following screenshot shows a successful logout request to the application. The area outlined in red shows the valid `laravel_session` session cookie. The area outlined in blue shows the date and time of the logout.



³ <https://auth0.com/docs/secure/tokens/json-web-tokens/json-web-token-claims>

The following screenshot shows a request to retrieve profile information on the /my-profile endpoint. The area outlined in red shows the same laravel_session session cookie as in the first screenshot. The area outlined in blue shows the date and time of the request. Note that the time is after the successful logout in the first screenshot, meaning the session cookie is still valid after logout.



```
Request
Pretty Raw Hex
1 GET /my-profile HTTP/2
2 Host: student.puritas-college-acc.com
3 Cookie: laravel_session=
  eyJpdiI6InVsY01SVWdpU0hHNWtNOUxIUmpPWWc9PSIsInZhbHV1IjoiaWFjUkQ1aTV3encwTUtNb3AwcEQrOT
  hmaFlnQlpxcG5ZOEdsMWpyUVpMNUEwcHdweUI4Nm5wcFBtRnhpKlFjRk42T2VGamVUR2g5NUpadmRwVnh2UEc4
  NEhEZUx1LO5sQkxyNO1QK2tZUmpQdmpDTW4wa3JFVUNTaDQ4THlob0YiLCJtYWMiOiIyZDRiMTY4NTY3ZTI3Mm
  IiMjIwOThiMzQwYWNlNjB1ODRkYzMiYjI4NzEzNDYyNWRhMDE4NDNkYWNiYjlmNDk3IiwidGFuIjoiaW043D;
4
5

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=UTF-8
3 Content-Length: 129060
4 Date: Tue, 08 Apr 2025 12:48:19 GMT
5 X-Amzn-Remapped-Date: Tue, 08 Apr 2025 12:48:19 GMT
6 X-Amzn-Requestid: 6b21ad83-9f1a-4756-8f68-8361ab7e97d2
7 Access-Control-Allow-Origin: *
```


5. Conclusion

5.1. Research questions

The following research questions were formulated prior to the test:

1. To which extent can a student gain unauthorized access to sensitive data from other students, such as certificates or course materials?
2. To which extent can a student start or follow a course after the subscription of that employee has expired?
3. To which extent can a student or manager gain unauthorized access to course material which is not explicitly assigned to these users?
4. To which extent can a manager manipulate data, such as altering or deleting students or courses?
5. To which extent can a manager gain unauthorized access to information about (students of) other organizations?

5.1.0. Overall answer to the research questions

During the penetration test, finding 4.1.1 was used to successfully take over the entire tenant environment of the Reducate application and consequently allow access to manipulate other students, courses, managers, etc. This was due to a mass assignment vulnerability within the edit profile form that allowed any user capable of modifying their profile through the Student UI to change and escalate their roles. As such, finding 4.1.1 provides a positive answer to all research questions defined for this pentest.

To avoid repetition, each individual research question will include only a brief reference to this critical finding without going into further details.

5.1.1. Access to sensitive data from other students as a student

During the penetration test, except for finding 4.1.1 no vulnerabilities were identified that would allow a student to gain access to sensitive information regarding other students. Separation of data between students is correctly implemented to rely on the session cookies and no methods were found to bypass this.

5.1.2. Start or follow a course after subscription has expired

During the penetration test, besides finding 4.1.1 one other vulnerability was identified that allows a student to start or follow a course after the subscription has expired. This is due to the course verification process depending on input that can be manipulated by an attacker (see finding 4.1.3). Furthermore, the specific APIs that load the course content do not verify whether a user is allowed to access the course content based on a valid subscription.

5.1.3. Access to unassigned course materials

During the penetration test, besides finding 4.1.1 one other vulnerability was identified that allows a student to gain unauthorized access to course material. This again is due to finding 4.1.3,

allowing a student to access any course material within the same tenant given the knowledge of a course ID and lesson ID.

5.1.4. Manipulating data of students as a manager

During the penetration test, besides finding 4.1.1 one other vulnerability was found that allows a manager to manipulate data (to a certain extent) of a student within the organization. This is due to a Cross-Site Scripting vulnerability found within the events page (see finding 4.1.2), which a manager can use to gain access to the application in the context of any student that visits a malicious events page.

5.1.5. Access to information about students of other organizations

During the penetration test, except for finding 4.1.1 no vulnerabilities were identified that would allow access to information (about students) of other organizations. Separation between organizations is correctly implemented to rely on session cookies and no methods were found to bypass this.

5.2. Remaining observations

5.2.1. Long expiration JWT token LRS

During the penetration test it was found that the JWT token used for the LRS API has an expiration time of 1 month. Combined with finding 4.1.4, which describes an insecure logout mechanism and results in any JWT token remaining valid even after a successful logout, this results in an increased attack surface and therefore risk regarding the use of these tokens.

5.2.2. Bypassing course process

The penetration test revealed that course progress tracking and answer verification are primarily implemented on the client side via ReactJS. This approach impacts the integrity of the course, allowing students to bypass all verification and manipulate both course progress and exam answers. While Reducate has acknowledged and accepted this as a known risk, it is important to note that this may have downstream implications for client organizations who depend on accurate course completion and assessment data for things like compliance, certification, or employee development tracking purposes.

5.2.3. Discrepancy between Student UI and other UIs

While the Student UI, Manager UI and Tenant UI are built on the same framework and provide somewhat similar functionality, it is important to highlight that most of the identified vulnerabilities were found exclusively within the Student UI while being properly mitigated within other UIs. This can be an indication that security within the Student UI may not be prioritized to the same extent as other UIs.

5.3. Open questions

All components were addressed during the penetration test.

5.4. Recommendations

Based on the above, The S-Unit recommends the following:

1. Apply the recommended changes for the individual findings, prioritizing those with the highest impact to efficiently minimize the risk
2. Investigate to which extent the lifetime of JWT tokens should be 1 month or whether this can be reduced to a shorter timeframe
3. Consider implementing verification of course progress and exam answers on the server side, thereby preventing students from manipulating courses
4. Investigate if and why security within the Student UI is notably different compared to other UIs and consider implementing the same, strict security checks across all UIs

Risk Classification of Vulnerabilities

Findings are summarized and classified into one of four classes: critical, high, medium, or low. Similar individual findings are combined in the management summary. During combination, the individual finding with the highest classification determines the minimum level in the reported classification. The effect of the combined findings on the assets, as defined by the client, is weighed into the final classification.

Critical

The finding leads to a direct threat to the client's assets (e.g. hijacking of critical information systems, leaking sensitive information or severe damage to means of production).

High

The finding leads to an indirect threat to the client's assets. Either because findings are related to non-critical components or because successful exploitation requires overcoming additional obstacles (e.g. exploitation may depend on user interaction or on certain random factors).

Medium

The finding leads to a limited direct threat to the client's assets. Successful exploitation allows only limited access to non-critical components or a small attack surface leads to a high threshold for abuse (e.g. Man in the Middle attack).

Low

The finding does not lead to a direct threat to the client's assets. However, these findings are potentially valuable for an attacker when additional attacks are executed.

Impact

Brute Force

Vulnerabilities that could result in attacks on a system by means of guessing information that can lead to authentication (e.g. usernames, passwords, authentication cookies). This can be achieved through using brute force methods, through dictionary attacks or with the aid of specialized algorithms.

Cross-Site Scripting

Vulnerabilities where a third party can manipulate content or behavior in the browser of a user without prior knowledge of the underlying system. Different Cross-Site Scripting related vulnerabilities are classified in this category, (e.g. script insertion and cross-site request forgery).

DoS (Denial of Service)

Vulnerabilities that could be abused for (over-)allocation of system resources (e.g. memory, processor, network, etc.). This may harm the availability of an application or infrastructure.

Exposure of Sensitive Information

Vulnerabilities that could leak information (e.g. documents, permissions, or data). This exposure can exist both locally as well as remotely.

Exposure of System Information

Vulnerabilities that could leak sensitive system information (e.g. version information, installation directories). Exposed information can be used to identify specific vulnerabilities for later exploitation.

Hijacking

Vulnerabilities that allow the capture of sessions or communication channels by other users or attackers (e.g. stealing a cookie).

Manipulation of Data

Vulnerabilities that allow non-authorized users or attackers to change data without access / authorization.

Privilege Escalation

Vulnerabilities that allow a user or attacker to attain more permissions than granted. Users must be able to retrieve higher/wider rights (e.g. administrator or root privileges).

Security Bypass

Bypassing or disabling security measures (e.g. a login screen or login check).

Spoofing

Vulnerabilities where an attacker can pretend to be something or someone else. This could be a user, people, or systems.

System Access

Vulnerabilities that lead to system access.

Unknown

Describes remaining vulnerabilities, security issues and weaknesses not described in one of the previous impact explanations or when impact and consequences are unknown and not described by the supplier (e.g. 0-day).

Disclaimer

© The S-Unit B.V.

All rights reserved. The information provided in this document is confidential and exclusively intended for Reducate Services B.V. . Use of this information by anyone other than strictly authorized persons of Reducate Services B.V. is prohibited. Disclosure, copying, distribution and / or distributing this information to third parties is not allowed. This document is subject to the General Terms and Conditions, deposited with the Chamber of Commerce. All trademarks mentioned in this document are the property of their respective owners.

The S-Unit | Your Security Companion