PHILIPS

Security Testing Report

Engage v6.5.3

PHM\Vital Health

Table of Contents

Document Version Control

Name of the document: Engage v6.5.3 Security Testing Report		
Version: 4.0	**Intake ID:**	2630
Document Definition: This document highlights the vulnerabilities currently existing in the application under scope. It also documents possible actions to be taken to reduce/eliminate the vulnerabilities.	**Document ID:**	PRHC/C40/SVN/85681
Author: Varsha Seetharam & Akash Hari	**Effective Date:**	09/May/2023
Reviewed by: Chaitra N Shivayogimath & Shabana Bagum		

Document History

Version	Date	Author	Section	Changes
0.1	20 Dec 2021	Chaitra N Shivayogimath	Complete	Initial Draft
0.2	20 Dec 2021	Ashwin K K	Complete	Addition & Review
1.0	21 Dec 2021	Shabana Bagum	Complete	Final Review
1.1	02 May 2022	Sreerag M	Complete	Initial Draft
1.2	03 May 2022	Shibija K	Complete	Addition & Review
2.0	04 May 2022	Pranati Mohanty	Complete	Final Review
2.1	21/Oct/2022	Shibija K	Complete	Initial Draft

2.2	21/Oct/2022	Sreerag M	Complete	Addition & Review
2.3	03/Nov/2022	Akash Hari	Complete	Addition
2.4	03/Nov/2022	Shabana Bagum	Complete	Addition & Review
3.0	04/Nov/2022	Pranati Mohanty	Complete	Final Review
3.1	05/May/2023	Varsha Seetharam & Akash Hari	Complete	Addition
3.2	05/May/2023	Chaitra N Shivayogimath	Complete	Addition & Review
4.0	08/May/2023	Shabana Bagum	Complete	Final Review
4.1	09/May/2023	Varsha Seetharam	8.1- Reduced the severity to Low 8.3 – Closed the finding	Addition & Review

Distribution List

User/Department/Stakeholder	E-Mail ID
Project Owner and PSO	corne.van.driel@philips.com, Benjamin.Williams@philips.com, Fleur.Maagdenberg@philips.com, freek.weijers@philips.com

1. Definitions & Abbreviations

Term	Explanation
SCoE	Security Center of Excellence
TLS	Transport Layer Security
SSL	Secure Socket Layer
XSS	Cross Site Scripting
HTML	Hyper Text Markup Language
VH	Vital Health
PHM	Population Health Management

The severity of every vulnerability has been calculated by using industry standard **Common Vulnerability Scoring System (CVSS)** used for assessing the severity of computer system vulnerabilities. CVSS provides a way to capture the principal characteristics of a vulnerability and produce a numerical score (Scores range from 0 to 10, with 10 being the most severe) reflecting its severity. The numerical score can then be translated into a qualitative representation (such as low, medium, high, and critical) to help organization properly assess and prioritize their vulnerability management processes.

The severity rating for the numerical values are mapped below:

None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

The **Severity** and **CVSS vector** of each vulnerability is calculated using the CVSS V3 **Base Score Metrics** Calculator located here. Vulnerabilities identified during security assessment are classified into standardized categories. Refer following table for more information:

Categories for vulnerability classification

Web application security assessment	OWASP Top Ten - 2021
Mobile application security assessment	OWASP Top Ten - 2016

IoT/Hardware security assessment	OWASP Top Ten – 2014

2. System Details & Architecture

Brief about the product architecture:

- Engage is a web application used for managing health related data of patients and primarily accessed by patients, General Practitioners (GP) and hospital professionals for recording different type of tests and its observations for the respective patient. Engage web application is running on IIS web server hosted on Windows server platform. The application is accessible externally from the internet.
- Application is developed on Microsoft .Net framework and uses jQuery JavaScript libraries.
- Testing environment: Test environment for Pen Test.

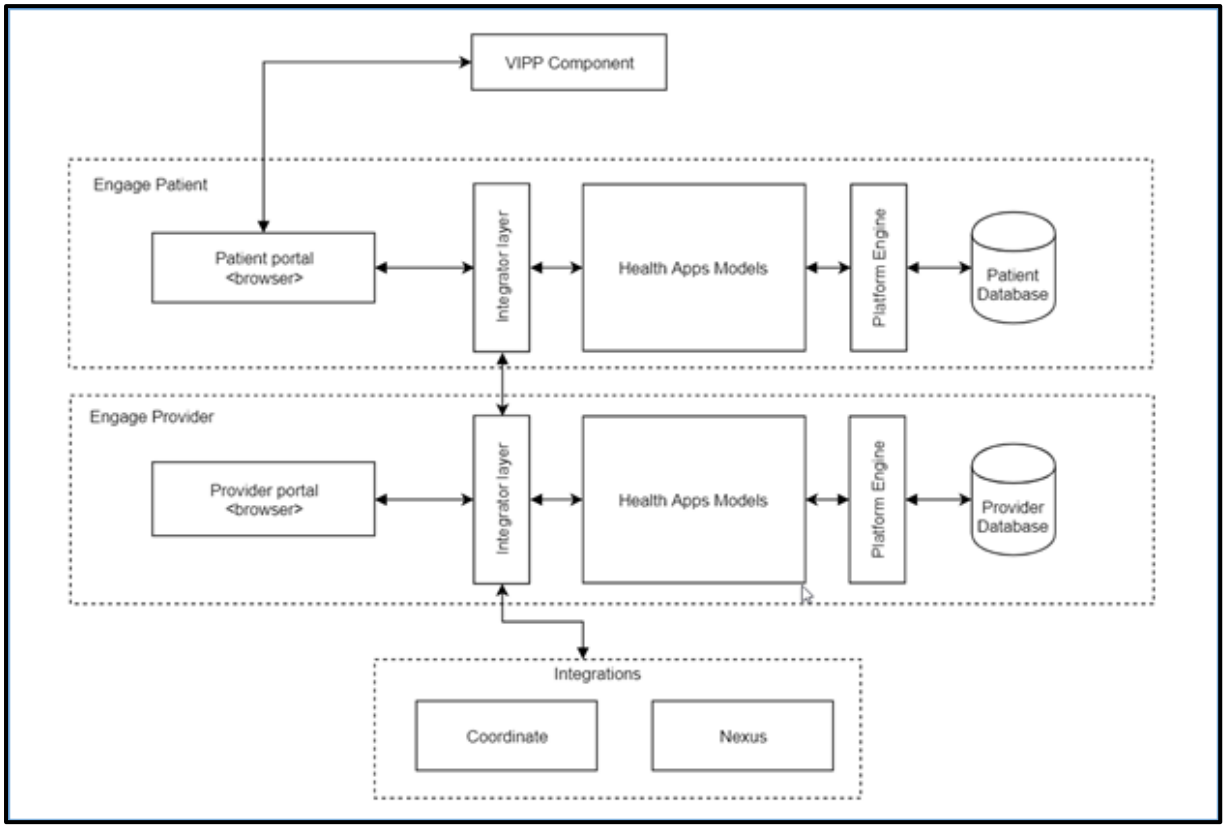


Figure 1: Architecture Diagram

3. Scope

The scope of this security assessment is to perform **Grey-Box** security testing to find security threats that may come from a malicious outsider or insider user of the **Engage v6.5.3.** Security testing on ***Web application, Android & iOS Mobile Applications*** of the **Engage v6.5.3.** is performed.

The following list includes some examples of major activities performed during the assessment:

Web Application:

- Crawl through complete scope of the web application/service space and identify for any unauthenticated URL or directory.
- Check for all input injection based attacks across all the possible entry fields in Web API.
- Exploiting any known component vulnerability or service misconfiguration.
- Reviewing the transport layer security implemented.

Android/IOS applications:

- Perform comprehensive "crawl" of all authenticated and unauthenticated application screens using test account.
- Applications was checked for local storage of the applications in mobile devices.
- Application transport layer protection is tested for clear text transmission of application data & for weak ciphers.
- Each application screen is tested to ensure that applicable authentication and/or authorization requirements are properly enforced along with associated business logic controls.
- Proof-of-concept exploits are performed, and applicable screenshots are captured to illustrate vulnerabilities.
- All application components are reviewed for conformance with GDS Application Security Directives.

Follow "Test case execution" section to get the detailed about test cases.

The test scope for this release is explained in the below table:

Start Date	End Date	Applications/Devices/IP's/URL's
19/April/2023	29/April/2023	Web URL: https://singleinstance-internalpentest.vitalhealthsoftware.com/ • Version: v6.5.3 • Environment: Pentest • User Role: Provider, Patient

		FHIR profiles.zip
24/Apr/2022	03/May/2022	Mobile App Name: • Android: Engage-2.5.3-ci316015-adhoc-signed.apk • IOS: Engage-2.5.3-ci316015-adhoc-signed.ipa • Version: v6.5.3 • Environment: Pentest User Role: Patient Mobile Application API services: Engage Mobile WebServices.zip

4. Not in Scope

- Portal Studio – Admin Login
- Source code review
- https://forcare0-consent.vitalhealthsoftware.com/ - Access Forbidden

Note: We have covered the testing of ***Engage v6.5.3*** in the environment provided and the results are valid if the same environment is replicated. Re-run the tests, if a new propagation of the environment is made.

5. Executive Summary

Security Center of Excellence (SCoE) team is engaged in activities to conduct security assessment of **Engage - v6.5.3** which included **Web Application, Android & iOS Mobile Applications** in scope. The purpose of the engagement is to evaluate the security of the **Engage - v6.5.3** against industry best practice criteria.

Note: Only highlights of important vulnerabilities are described below. Please refer 'Vulnerability Summary' section for complete detailed list of vulnerabilities.

During the security assessment following factors were found with consideration for significant improvement:

- Authentication Token in URL
- Broken Access Control

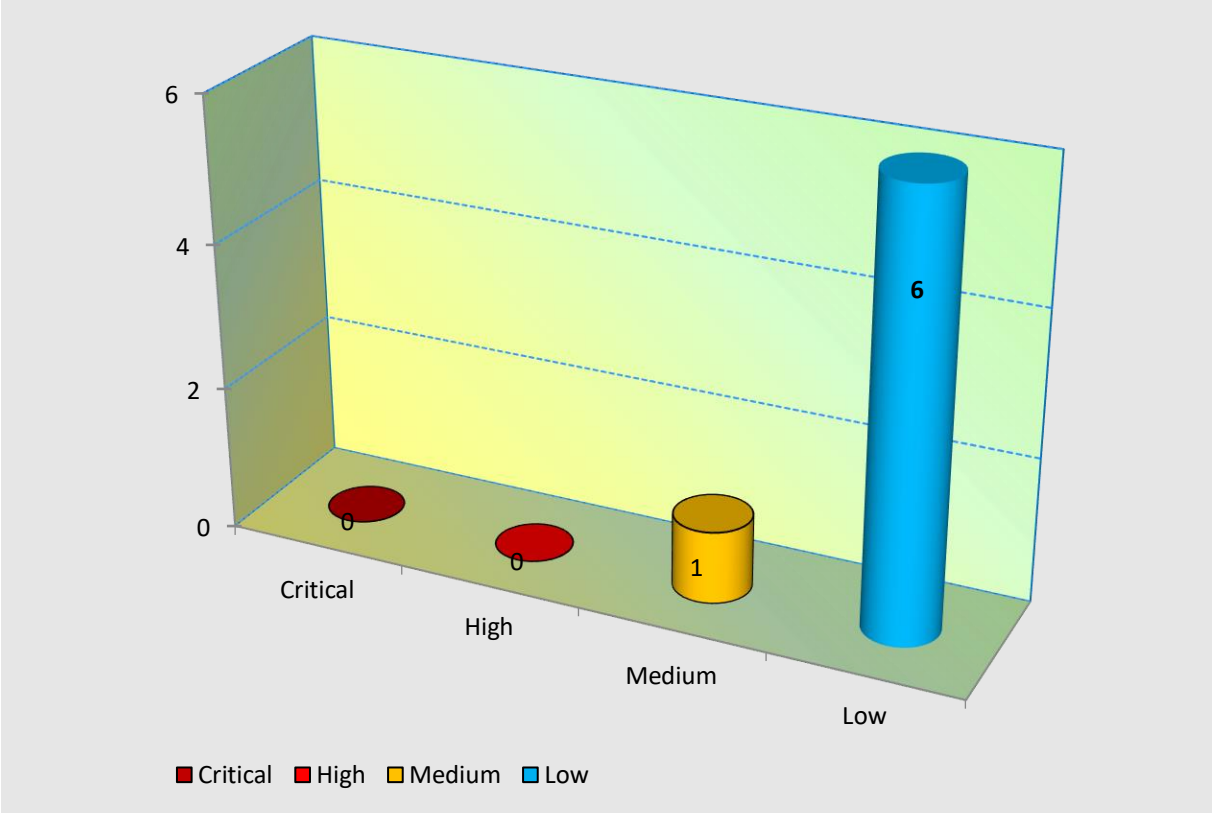
During the security assessment, security issues in the below areas are not found:

- Database Injection
- Cross Site Request Forgery attacks.

VULNERABILITY SUMMARY CHART

The graph below shows a summary of the number of vulnerabilities and their severities.

Note: The vulnerabilities mentioned in this report are technical vulnerabilities only. The Product Security Risk Assessment would report the business risks associated with these vulnerabilities.



6. Vulnerability Summary

The findings and vulnerabilities from the assessment are explained in the below table:

Finding No.	Vulnerability Title	Technical Risk	Impacted Area	CVE ID*	Status
86202	Unrestricted file upload	Low	WebApp	NA	Open
76575	Improper Authorization	Medium	WebApp	NA	Open
76339	Sensitive information in URL	Medium	WebApp	NA	Closed
86410	Improper Input Validation	Low	Webservices(mobile)	NA	Open
86200	CSV Injection	Low	WebApp	NA	Open
86201	Verbose error message	Low	WebApp	NA	Open
86203	Unsername Enumeration	Low	WebApp	NA	Open
35843	Input returned in response	Low	MobileApp - Both	NA	Open
81693	HTTP TRACE Method Enabled	Low	Webservices (mobile)	NA	Open
23049	Stored HTML Injection	Informational	WebApp	NA	Open
53228	Insecure Local Storage	Informational	MobileApp - Android	NA	Open

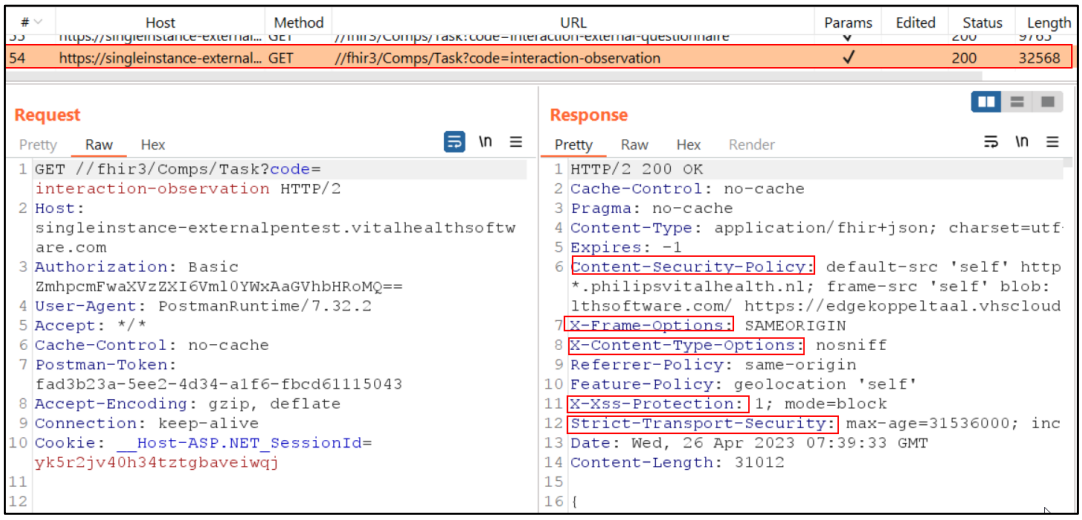
*CVE ID are mentioned for the vulnerabilities which has a known external CVE.

7. Observations

Below mentioned observations are not considered as vulnerability but informative to the business.

Observations which shows good implementation or best practice identified

- FHIR API's are set with most of the security headers.



- Strong cipher suites are supported by the server and uses TLSv1.2 protocol for secured connection.

```
nmap -sV -p - -Pn --script ssl-enum-ciphers singleinstance-externalpentest.vitalhealthsoftware.com
Starting Nmap 7.91 ( https://nmap.org ) at 2023-04-26 13:32 India Standard Time
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Nmap scan report for singleinstance-externalpentest.vitalhealthsoftware.com (3.124.245.161)
Host is up (0.19s latency).
rDNS record for 3.124.245.161: ec2-3-124-245-161.eu-central-1.compute.amazonaws.com
Not shown: 65534 filtered ports
PORT    STATE SERVICE   VERSION
443/tcp open  ssl/https
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-trane-info: Problem with XML parsing of /evox/about
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (ecdh_x25519) - A
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 2048) - A
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 2048) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 2048) - A
|     compressors:
|       NULL
|     cipher preference: server
|_  least strength: A

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 461.28 seconds
```

- Allow backup flag is set to false.

```
<uses-permission android:name="android.permission.USE_BIOMETRIC"/>
<application android:allowBackup="false" android:appComponentFactory="androidx.core.app.CoreComponentFactory" android:debuggable="false" android
:extractNativeLibs="true" android:icon="@mipmap/ic_launcher_square" android:label="@string/app_name" android:name="
crc641fe6c62fd0b8a458.CurrentActivityResolver" android:theme="@style/Theme.App.Starting">
    <receiver android:exported="true" android:name="com.google.firebase.iid.FirebaseInstanceIdReceiver" android:permission="
    com.google.android.c2dm.permission.SEND">
```

Observations which shows missing best practice or possible weak implementation (this may/may not be direct active threat):

- Cache Control header in FHIR API's is set to no cache which is consumed by the web application stores the sensitive information in the browser.

#	Host	Method	URL	Params	Edited	Status	Length
54	https://singleinstance-external...	GET	//fhir3/Comps/Task?code=interaction-observation	✓		200	32568

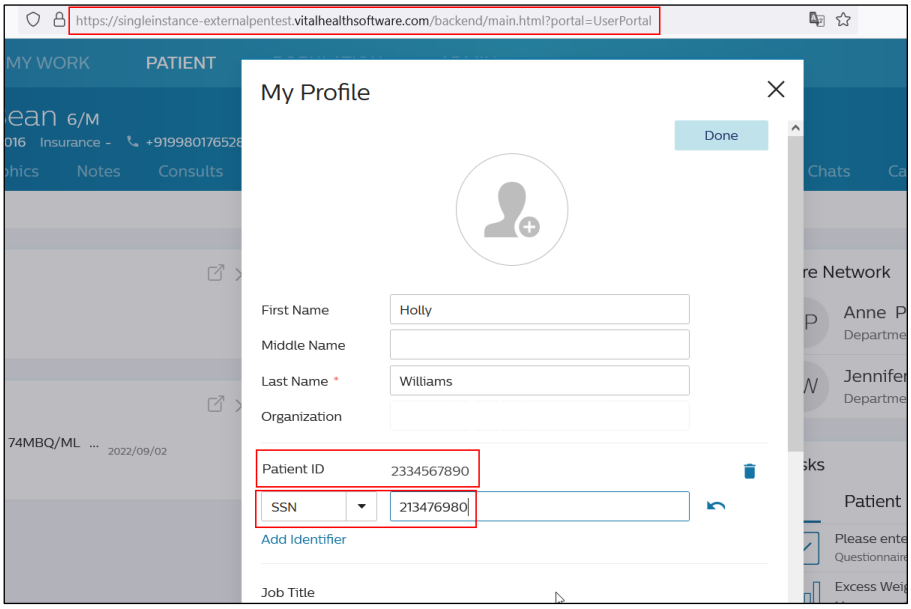
Request

```
GET //fhir3/Comps/Task?code=
interaction-observation HTTP/2
Host:
singleinstance-externalpentest.vitalhealthsoftw
are.com
Authorization: Basic
ZmhpcmFwaXVzZXI6Vml0YWxAaGVhbHRoMQ==
User-Agent: PostmanRuntime/7.32.2
Accept: */*
Cache-Control: no-cache
Postman-Token:
fad3b23a-5ee2-4d34-a1f6-fbcd61115043
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cookie: __Host-ASP.NET_SessionId=
yk5r2jv40h34tztgbaveiwqj
```

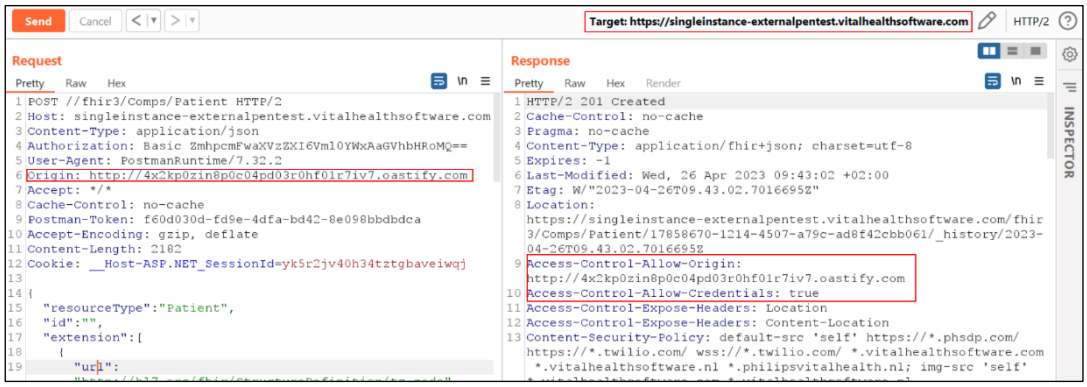
Response

```
HTTP/2 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/fhir+json;
charset=utf-8
Expires: -1
Content-Security-Policy: default-src 'self'
https://*.phsdp.com/ https://*.twilio.com/
wss://*.twilio.com/ *.vitalhealthsoftware.com
*.vitalhealthsoftware.nl
*.philipsvitalhealth.nl; img-src 'self'
*.vitalhealthsoftware.com
*.vitalhealthsoftware.nl
*.philipsvitalhealth.nl data:; object-src
blob:; script-src 'self'; style-src 'self'
*.vitalhealthsoftware.com
*.vitalhealthsoftware.nl
```

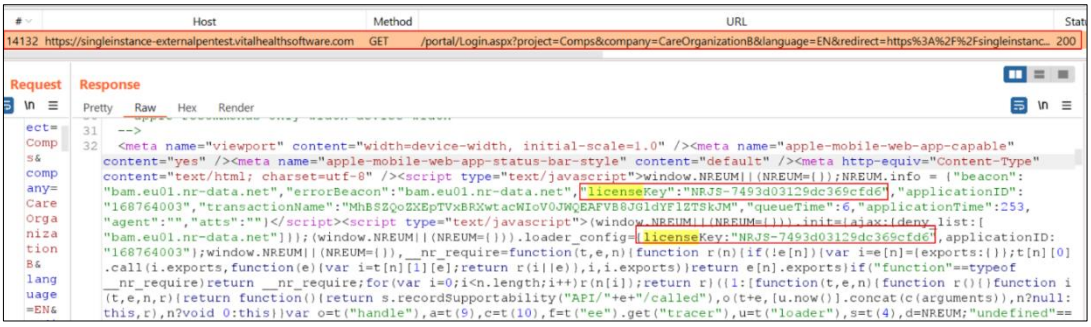
- It is observed that the application does not mask NPI data such as SSN.



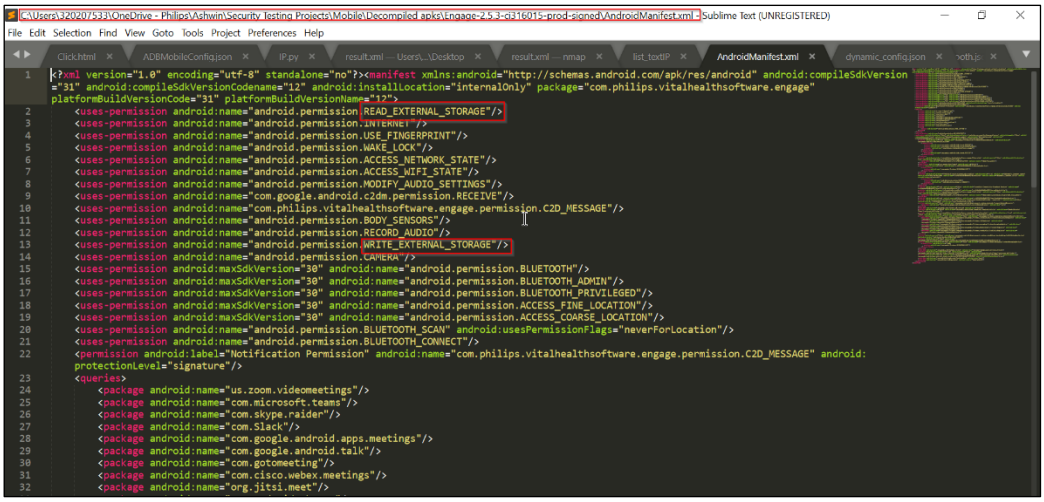
- During the assessment, it is observed that the HTTP Response header Access-Control-allow-origin takes the Origin from the request which means any domain can request and server would respond to that call normally considering it as legitimate. Ideally, it should allow only permissible origins.
  Access-Control-Allow-Credentials: true is considered insecure.
  An unrestricted CORS policy allows an attacker to access sensitive data or perform unauthorized user actions without user knowledge. Reporting it as an observation, as there is no browser involvement now.



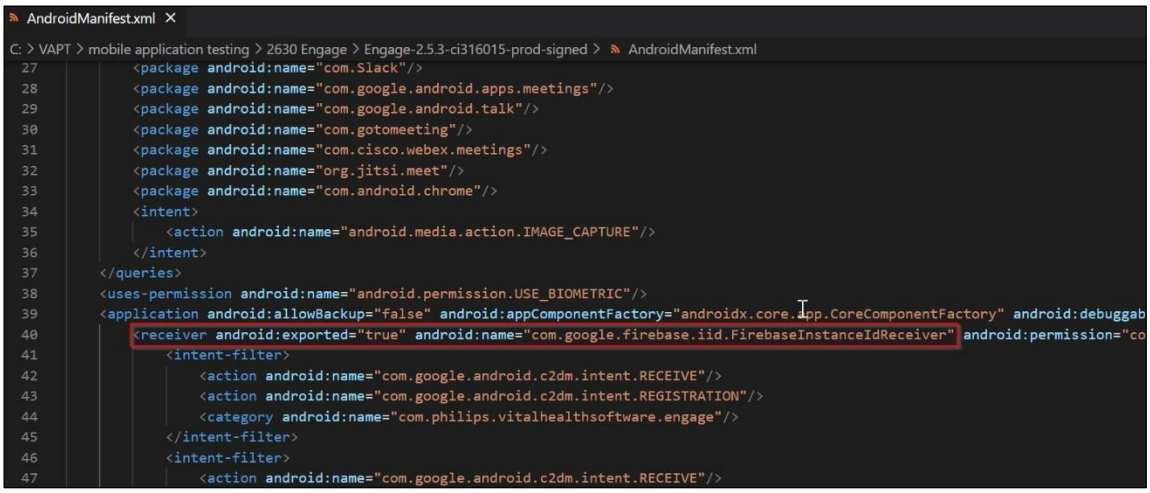
- It is observed during the testing that a License Key is revealed in the HTTP responses. Unsure of the usage of this License key this is reported under observation. Requesting the product team to check on this and act to not send the License Key in response if it's not required to.



- During the security assessment, it is observed that the application has many permissions like READ and WRITE to external device, which are considered dangerous as it allows more control over the device. The android application permissions are found in the Androidmanifest.xml file. There are many other permissions, which are also to be looked in for a safer side.



- During the security assessment of the product, it is observed that the Android application exports the following components for use by other applications but does not properly restrict which applications can launch the component or access the data it contains.
  - com.google.firebase.iid.FirebaseInstanceIdReceiver
  - com.google.android.gms.auth.api.signin.RevocationBoundService



8. Detailed Vulnerability Report

8.1 WebApp: Unrestricted file upload

Vulnerability Title	Unrestricted File Upload
Vulnerability Category	A5 – Security Misconfiguration
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.1 CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:L
Description	**Vulnerability Description:** If a file upload functionality within an application allows the user to upload any file without any restriction on the file type, the server becomes vulnerable to unrestricted file upload. Uploaded files may pose a significant risk if not verified and handled correctly. Attacker can upload malware/plant backdoor via this file upload feature. During the security assessment it was observed that the application has functionality to Import data as part of the Data tab. It was observed that using double extension malicious files were able to be uploaded. Once uploaded, we got the success response. **Revalidation(9th May, 2023):** According to the application team, Engage has checks on the file extension, and has a virus scanner running when uploading files. However, They have indeed no check on the contents of the file. As there is no integrity risk as the uploaded file cannot be changed. Also, when downloading the file again, and try to execute it in Windows, it is not executing based on the file contents but on the file extension. Hence the severity is reduced to Low. **Reference:** https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload **Exploitability rational:** An attacker should have some privilege role to upload the files.

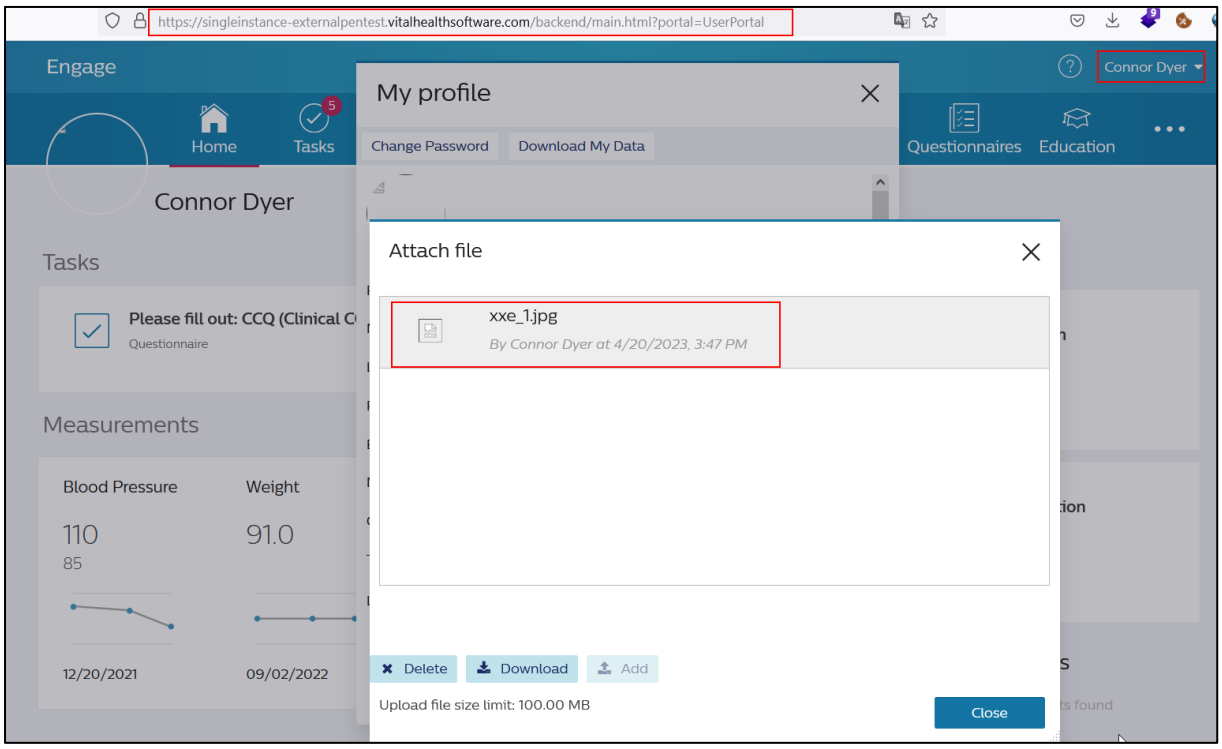
	Impact rational: An attacker could exploit this vulnerability by uploading a malicious file which can allow him to execute various attacks like upload virus, introduce pages vulnerable to vulnerabilities like XSS or worst case execute arbitrary code on the server.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/backend/submit-upload.html
Recommendation:	The application should validate uploaded files for type and size, and limit how often the user is able to perform uploads. The following validation should be performed: • If the application requires uploaded files to be of a specific type such as PDF, text, or Word Document, the application should validate that the extension is '.pdf', '.txt' or '.doc'. • The first four bytes of the file should be validated. These first few bytes are known as the file's 'Magic Number' and will uniquely identify the file type. For example all PDF files start with the byte-sequence '%PDF'. • An upper limit on file size should be enforced, as determined on a case-by-case basis. For instance, if a typical file upload is 10 MB, the application should reject files that are larger than 25 MB. • The frequency of file uploads should be validated. If the application detects a high frequency of file uploads from a single user, the application should prohibit the user from uploading files for a period of time. • Contents of the file also need to be validated. MIME type can be checked for mitigation. In addition to the primary criteria above, all uploaded files should be scanned for known malware/viruses. Reference: https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html
Status	**Open**

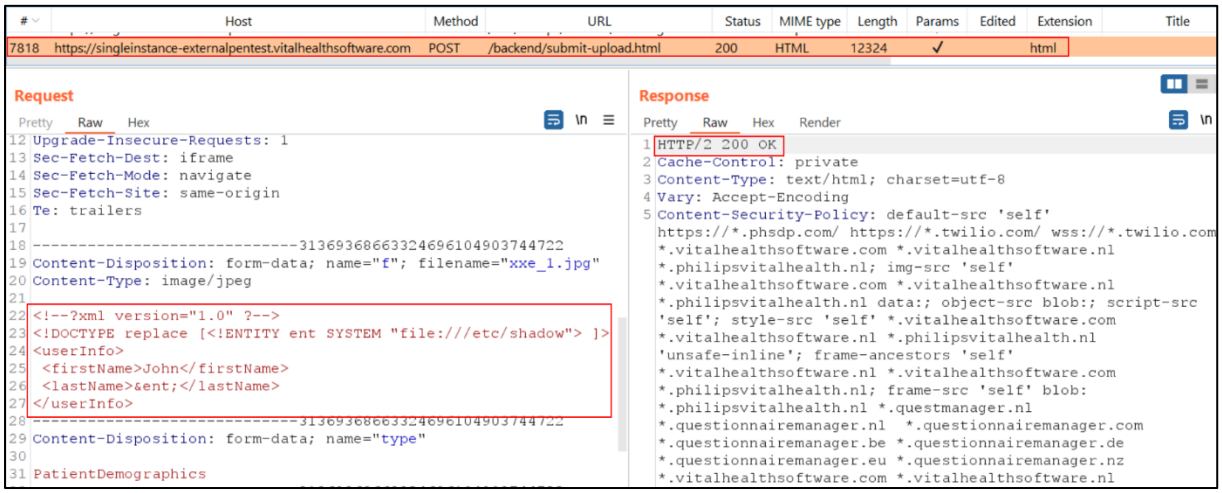
Steps to Reproduce:

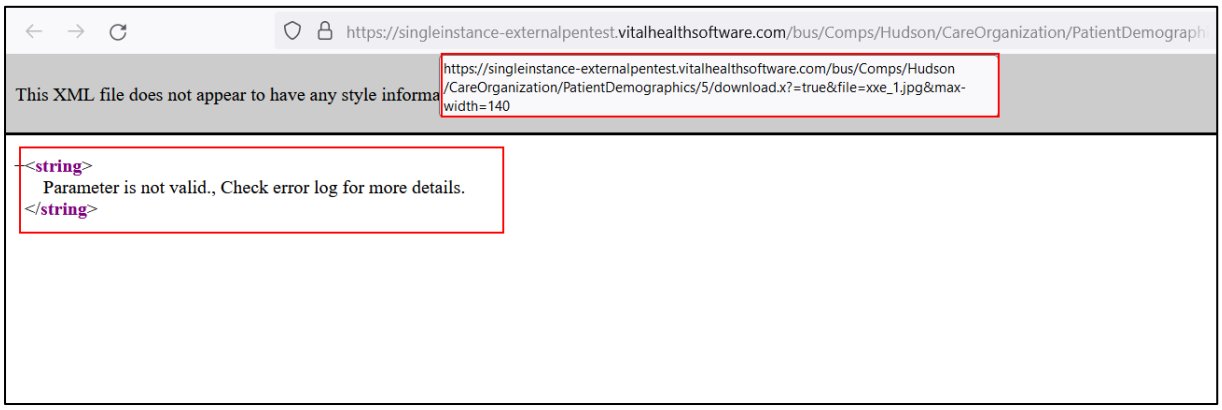
1. Login to the application.
2. Go to the profile icon and upload the profile picture with XML content.

3. Observe that the application does not check for file content.
4. Profile picture successfully gets uploaded.
5. Also, you can observe that the provider who can access the impacted patient profile can also be affected by this issue.

Supportive evidence:







8.2 WebApp: Improper Authorization

Vulnerability Title	Improper Authorization
Vulnerability Category	A1 – Broken Access Control
Severity	**Medium**
CVSS V3 Calculation	CVSS Base Score: 5.3 CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:N/A:N
Description	**Vulnerability Description:** During the security assessment, it is found that, the check for access control was missing for multiple backend types. Improper Access control allows an unprivileged user to execute actions and retrieve information which they are not supposed to do. **Exploitability Rationale:** Any valid user of the application can exploit the issue. **Impact Rationale:** Unauthorized access to application resources.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/api/data/Comps/SMD.Common.ProgressStatuses
Recommendation	It is recommended to verify user role at both client & server side and allow operations only for valid user roles.
Status	**Open**

Steps to Reproduce:

1. Login to the application.
2. Access the url - https://singleinstance-externalpentest.vitalhealthsoftware.com/api/metadata/Backend.Types/fingerprints-311c51de427cf72592f9fd42498392eb443ed2244143a7cf246832d39a15b975
3. The Backend URL's will be listed.
4. Try to access each URL.

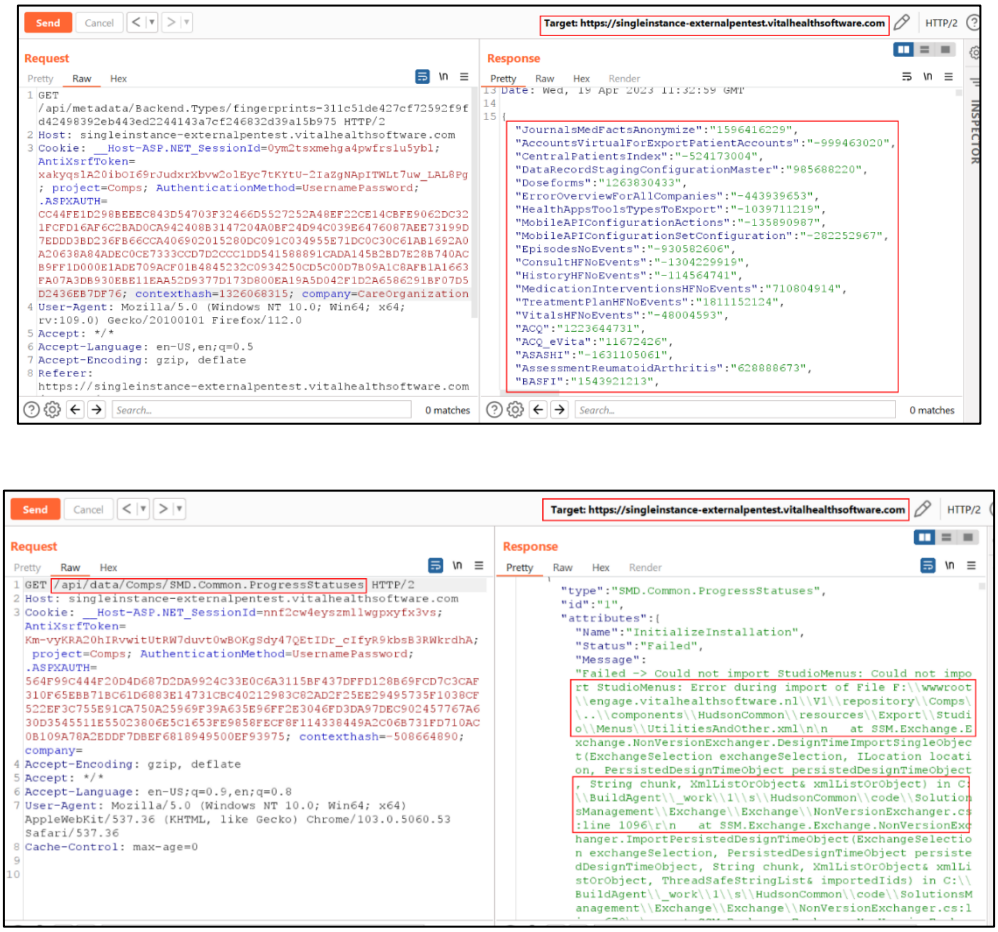


Figure 1: Provider Portal – dramory user

Note: Below attached is the output of access check to all backend types which consist of Backend type and Response code.

Patient Portal	Provider Portal
Cdyer Patient_cdyer.xlsx	ejackson Provider_HCP - ejackson.xlsx

8.3 WebApp and Webservices: Sensitive information in the URL

Vulnerability Title	Sensitive information in the URL
Vulnerability Category	A02: Cryptographic Failures
Severity	**Medium**
CVSS V3 Calculation	CVSS Base Score: 4.2 CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:L/A:N
Description	**Vulnerability Description:** During the assessment, we observed that the jwt token is exposed in transit between the client and the server via URL query string parameter. URLs may be stored or viewed in multiple places during and after a request is made to the server: • If the URL is requested by clicking a link or manually entering the address, the query string is seen in the browser address bar. • URLs are often logged in multiple places including the browser history, proxy logs, and web server logs. • The query string is sent as a part of the URL if the URL is passed to another site via the referrer header. • URLs sent to the user as part of an HTML page may be cached on disk. **Revalidation (9th May, 2023):** The XDS Consent app authentication token in url issue has been reported before and has been closed by SCoE based on the justification given to the team. Also, regarding the FHIR API querystring which may contain sensitive fields (e.g. phone number, email address). An external service can search based on those fields via the querystring, this is default behaviour of the HL7 FHIR standard. So, this finding will be closed based on the justification given by application team. **Exploitability Rational:** Any attacker who gains access to any of the location where URLs are stored can view sensitive information passed via the query string. Potential access vectors may include but are not limited to: • Browser history, proxy logs, web server logs, etc.

	• Utilizing other attacks (such as cross-site scripting) to extract sensitive information from the source of a page containing links to URLs with sensitive information in the query string. • Shoulder-surfing the URL in a user's browser address bar. **Impact Rational:** The attacker can get access to authentication token which leads to unauthorized access to victim resources.
Affected Systems/IP Address/URL	https://forcare0-consent.vitalhealthsoftware.com/consent-app/?locale=en-US&jwt=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOiIyNzYxNDE4Yi0zZjY5LTRhZjAtOGQ0ZC1hZDllMWRhYjQyODAiLCJpYXQiOjE2ODE4MTY2NzUsImV4cCI6MTY4MTgyMDI3NSwibGFuZyI6ImVuLVVTIiwiZXh0ZW5zaW9ucyI6eyJpaGVfYnBwYyI6eyJwYXRpZW50X2lkIjoiNjkyMTUzOTAxXl5eXHUwMDI2MS4zLjYuMS40LjEuMjEzNjcuMjAwNS4zLjdcdTAwMjZJU08ifX19.-NkV6e7-2IuXd3nKyWo9Jj6EgyDL5zzJFyZ7lBTRDko https://singleinstance-externalpentest.vitalhealthsoftware.com//fhir3/Comps/Patient?email=test1@vitalhealthsoftware.com https://singleinstance-externalpentest.vitalhealthsoftware.com//fhir3/Comps/Patient?phone=+919987945386 https://singleinstance-externalpentest.vitalhealthsoftware.com//fhir3/Comps/Patient?birthdate=1945-02-28
Recommendation	Never pass the sensitive information between the client and server via URL query string parameters. Instead, the server should create and store the session identifier and then set it in a cookie on the client.
Status	**Closed**

Steps to Reprtoduce:

Instance 1:

1. Login to https://singleinstance-externalpentest.vitalhealthsoftware.com/ using patient user credentials.
2. Navigate to Consent link on top right dropdown list.
3. Click “Consent for external documents” link.

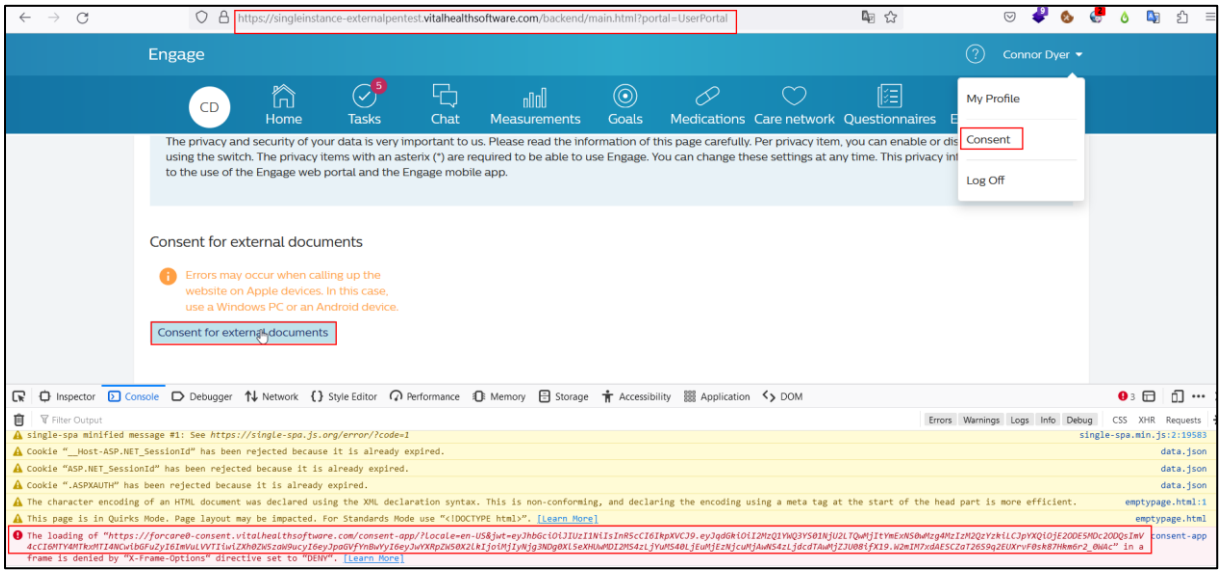


Figure 2: JWT token passed via URL

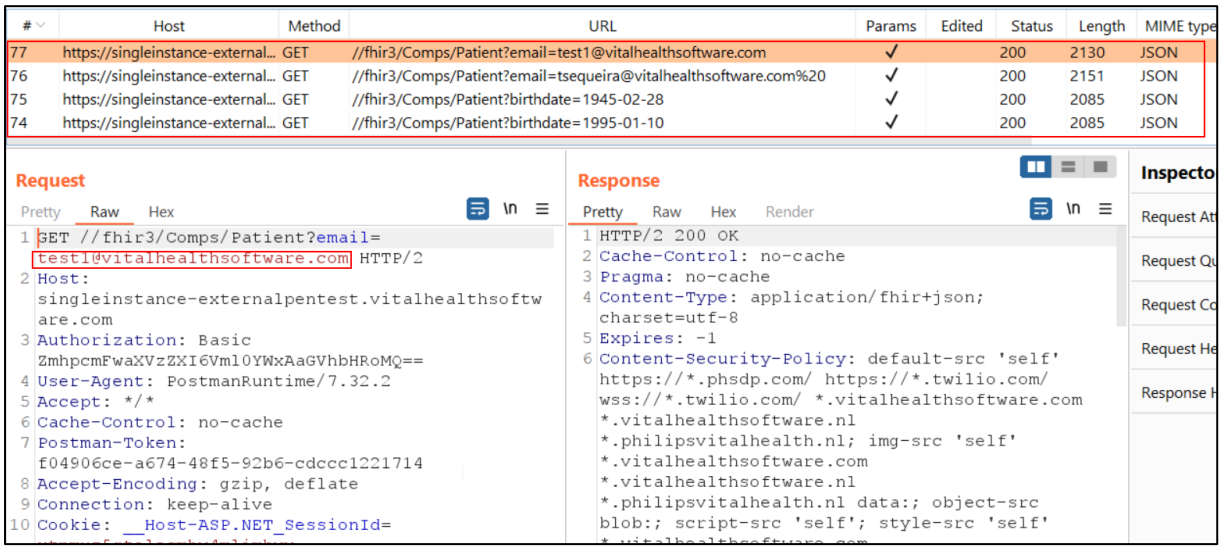
Note: The JWT token implementation was using HS256 algorithm. This is considered as a weak algorithm.

https://auth0.com/blog/brute-forcing-hs256-is-possible-the-importance-of-using-strong-keys-to-sign-jwts/

Figure 3: Usage of HS256 algorithm

Instance 2:

1. Access the API's using the FHIR credentials.
2. Observe that the most of the URL's contains sensitive information such as mail address, birth date in the GET request.



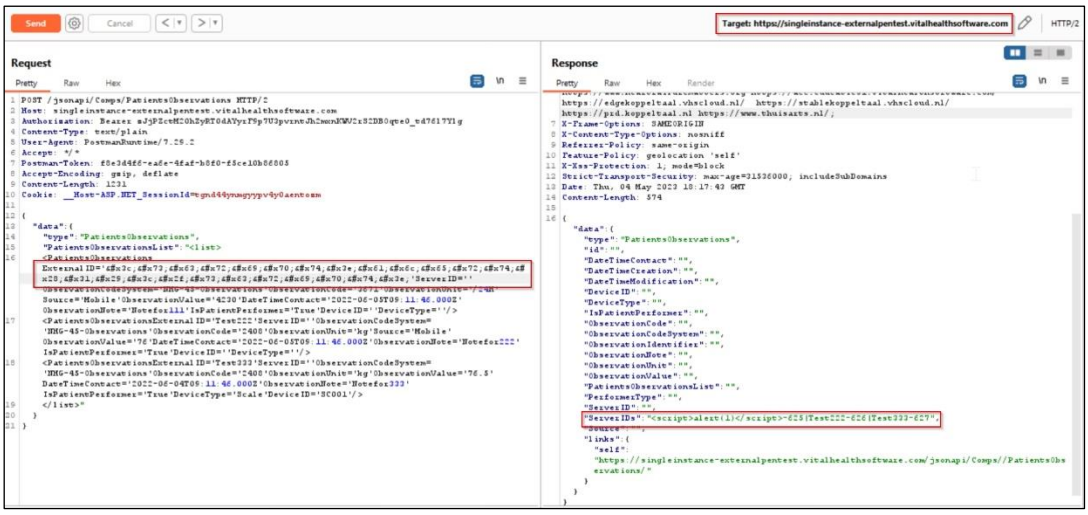
8.4 Webservices(mobile): Improper Input Validation

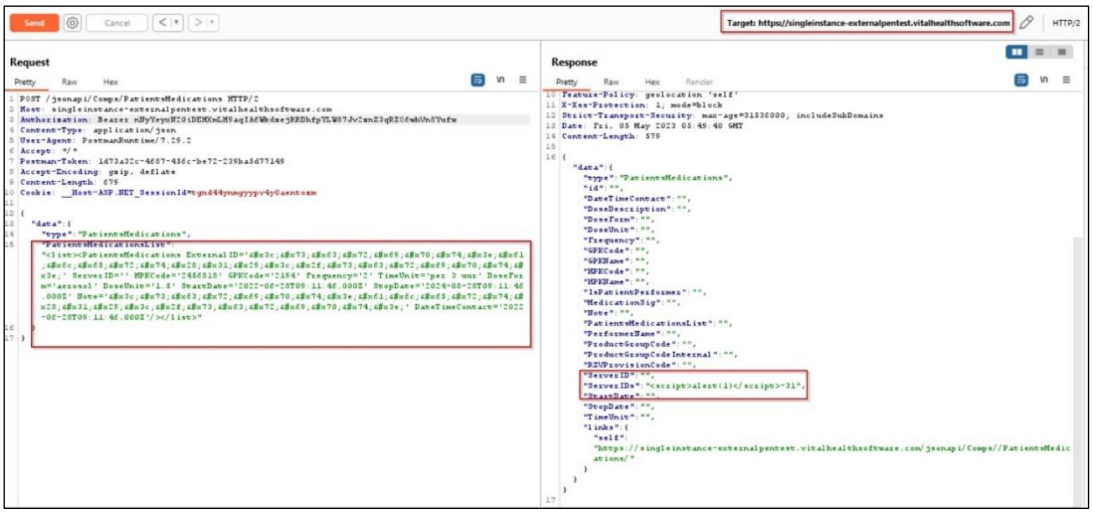
Vulnerability Title	Improper Input Validation
Vulnerability Category	A3 - Injection
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.5 CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N
Description	**Vulnerability Description:** During the security assessment of the product, it is observed that some of the APIs allows the usage of special characters and is accepting them while creating/updating several parameters. The same isn't validated at the client end. Due to this, the application may be vulnerable to attacks like XSS, SQL injection etc. The user-controlled input is not properly sanitized/validated. **Exploitability Rational:** The attacker needs to be an authenticated user. Failure to properly validate and handle untrusted input represents the single largest category of software security weaknesses. At a minimum, data that is not validated may impact the application's control flow or data flow, leading to unexpected application states for end users, unintended changes to back-end data, as well as unexpected outcomes from executed application logic. **Impact Rational:** An attacker may submit payloads that seek to exploit any number of vulnerabilities that typically result from a lack of input validation. These include (but are not limited to) SQL injection, cross-site scripting, LDAP injection, log injection, and command injection.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/jsonapi/Comps/PatientsObservations https://singleinstance-externalpentest.vitalhealthsoftware.com/jsonapi/Comps/PatientsMedications

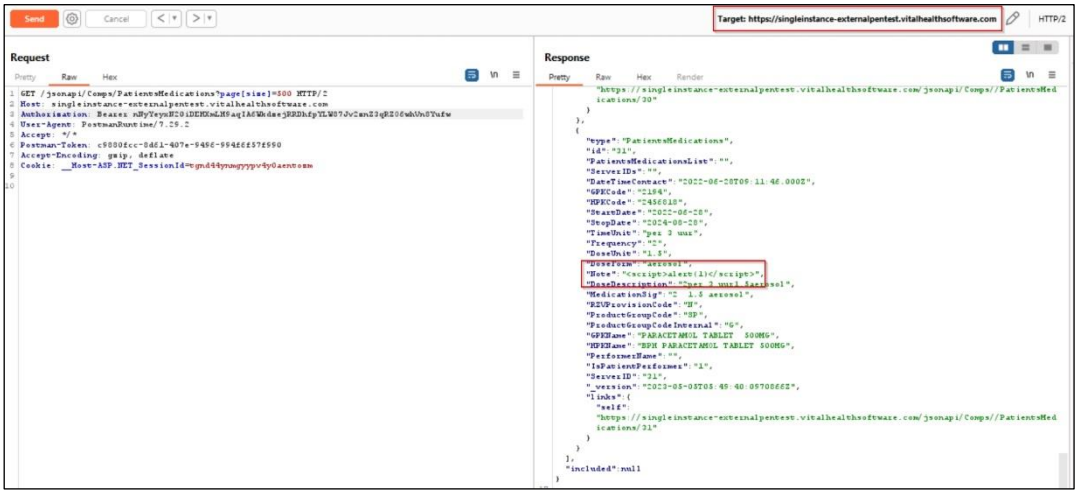
Recommendation	Implement input validation at the client as well as server side. This can be achieved by whitelisting. You can define what is allowed as input and reject everything else. It is recommended to implement checks for range, length, format and type of data.
Status	**Open**

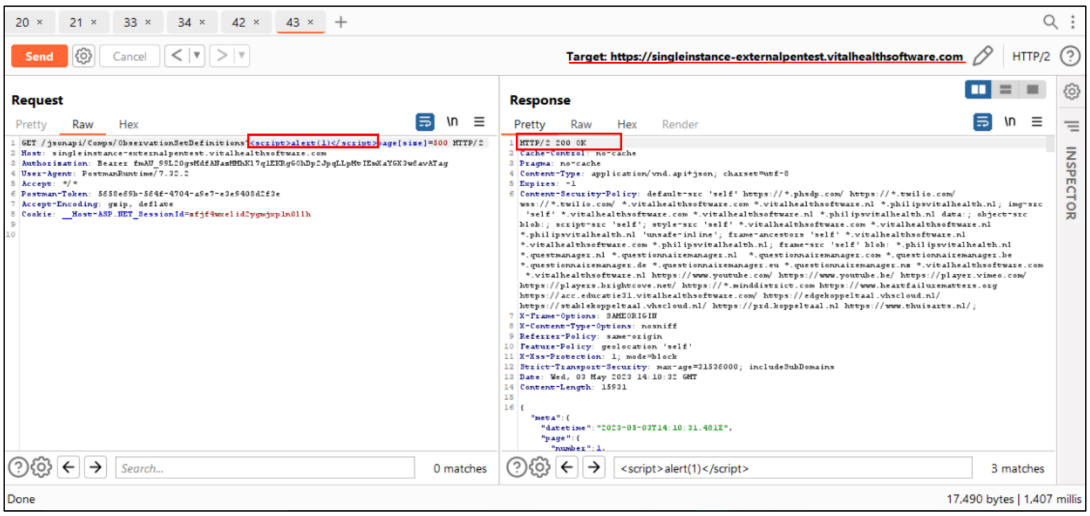
Steps to Reproduce:

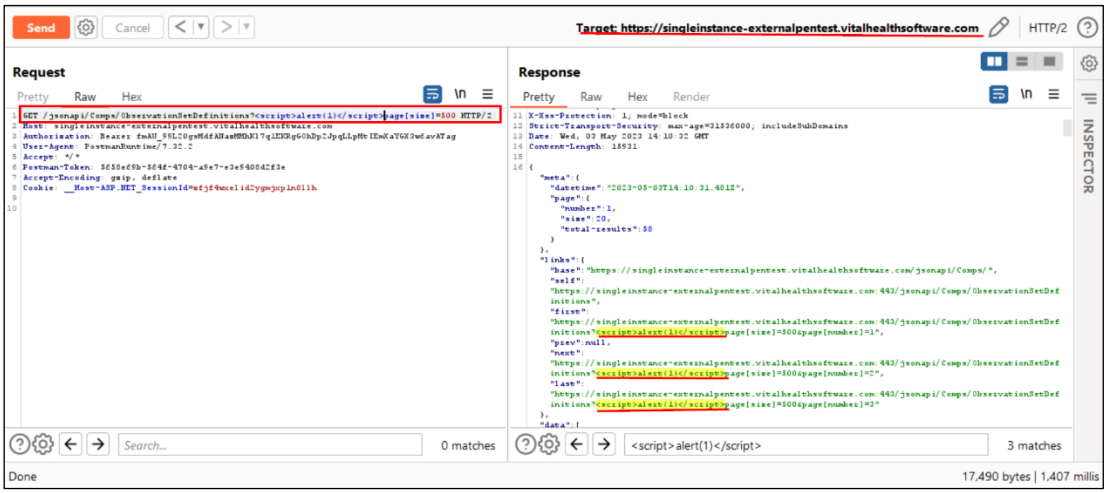
1. Configure postman to use proxy tool such as burp suite.
2. Capture affected endpoints and intercept the request.
3. Insert any special characters/javascript in html encoding, it is observed that the injected script is returned in the server response as shown below:











8.5 WebApp: CSV injection

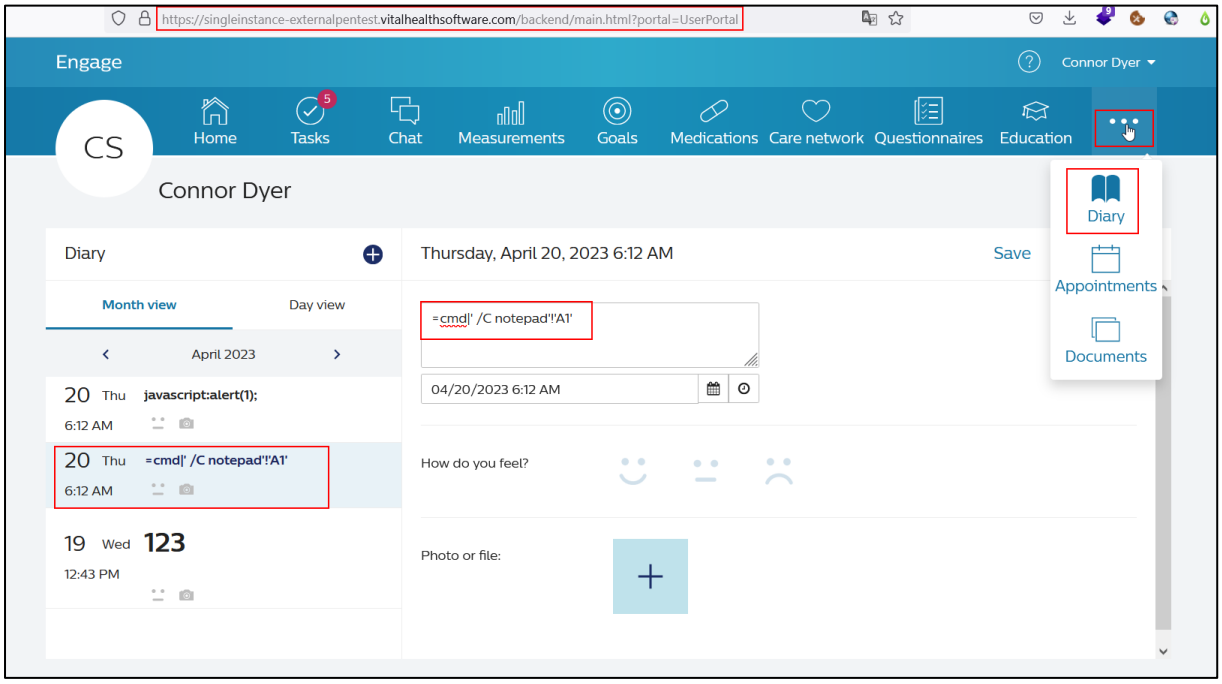
Vulnerability Title	CSV Injection
Vulnerability Category	A3 – Injection
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.7 CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:U/C:L/I:L/A:N
Description	**Vulnerability Description:** During the security assessment it was observed that the application has functionality to export data in XLS spreadsheet which is vulnerable to formula injection. Any user shall be able to inject the malicious payload as part of Input Fields while Editing and insert CSV Formulae. User can download his/her profile in Excel or CSV format. **Note:** There are multiple input fields of this issue. It is recommended to apply the fix across the application. **Exploitability Rationale:** Any valid user of the application can exploit the issue. **Impact Rational:** An attacker may inject functions or expressions that alter an affected spreadsheet's content to trick a victim into believing that the modified spreadsheet content is genuine. The impact of altering this data is contingent on what data is present in the document, but the overall goal would be to influence a victim's actions based on the modified data (e.g. altering market data to influence a victim's financial decisions). Additionally, an attacker may inject functions such as the =HYPERLINK (…) function to trick the victim into navigating to an attacker-controlled site or launching an executable on their local system, potentially leading to information leakage or complete system compromise.
Affected Systems/IP Address/ URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/bus/Comps/Hudson/CareOrganization/PatientPersonalDataExports/4/download.x?file=PatientPersonalData_2023-04-20_08-16-34.zip

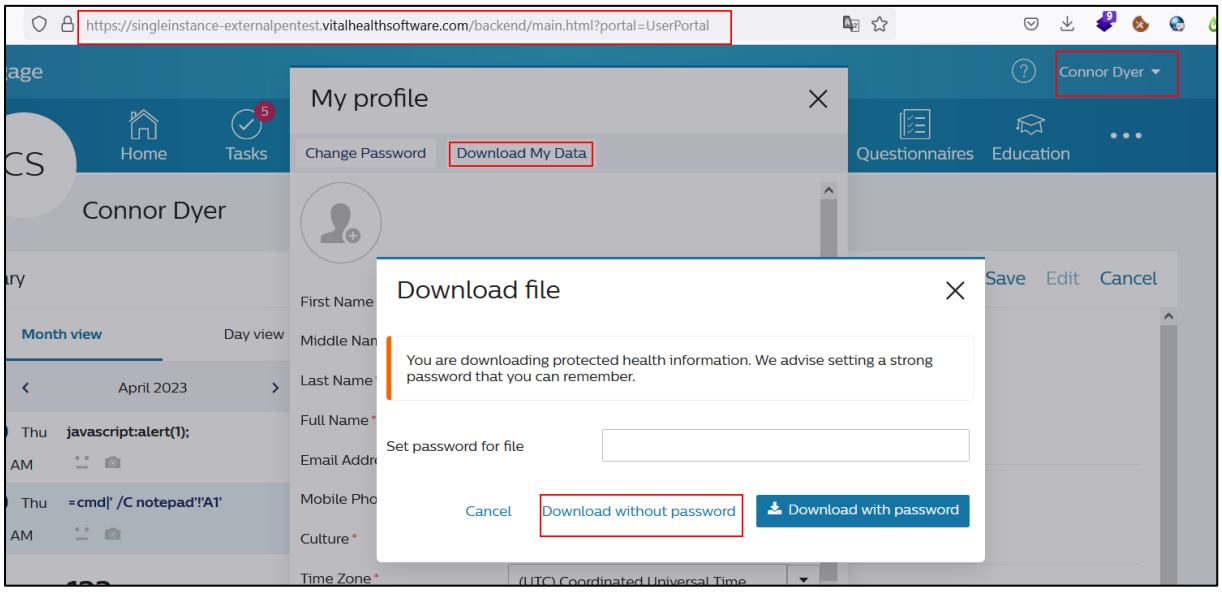
Recommendation	Escape all untrusted input before inserting it into spreadsheet data fields. In Microsoft Excel, this is accomplished by placing a single quote before the content. For example, the following string will be treated as plain text rather than a formula: '=HYPERLINK(…)
Status	**Open**

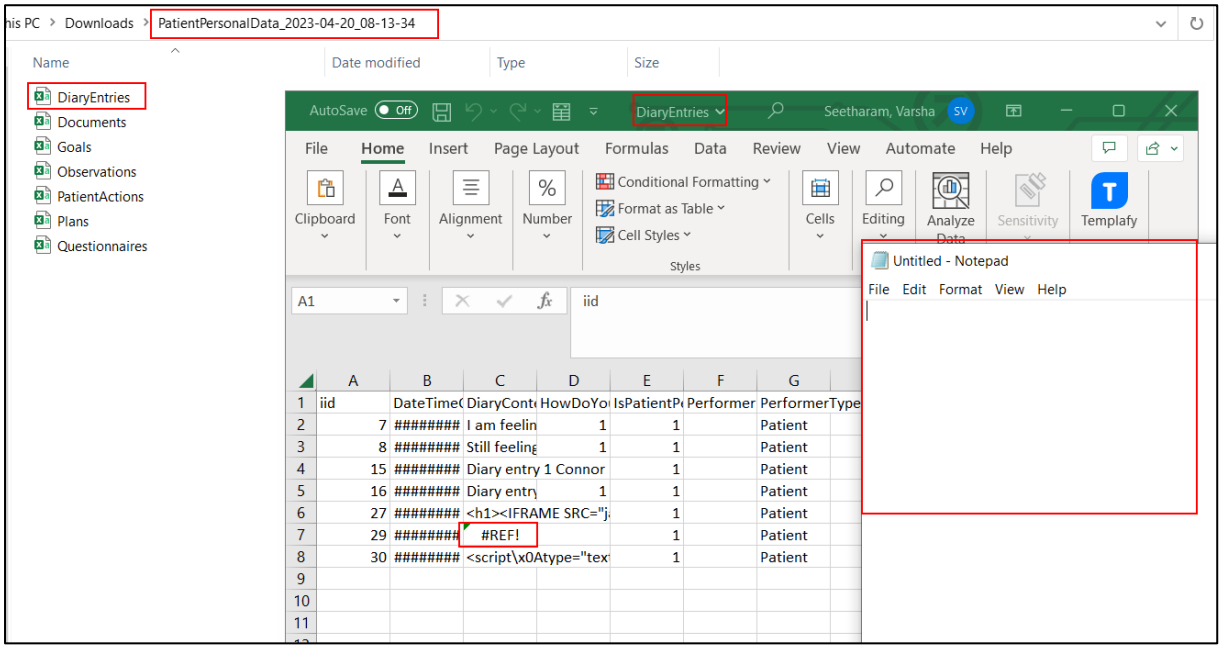
Steps to Reproduce:

1. Login to the application using any user.
2. Enter any profile data.
3. Here, we have entered diary details with formula injection payload.
4. Then download the profile without password.
5. Observe that the CSV gets downloaded and the formula payload gets executed.

Supportive evidence:







8.6 WebApp: Verbose error message

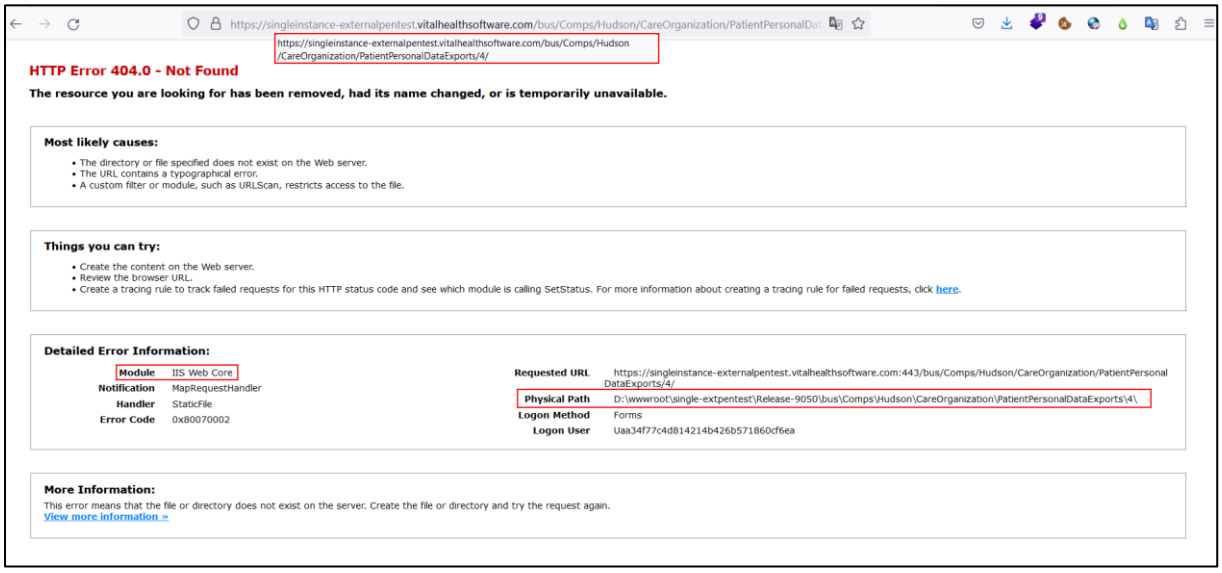
Vulnerability Title	Verbose error message
Vulnerability Category	A5 – Security Misconfiguration
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.4 CVSS:3.1/AV:A/AC:H/PR:L/UI:R/S:U/C:L/I:L/A:N
Description	**Vulnerability Description:** Improper handling of errors can introduce a variety of security problems for a web site. The most common problem is when detailed internal error messages such as stack traces, database dumps, and error codes are displayed to the user (attacker). These messages reveal implementation details which are supposed to be hidden. Reference: https://owasp.org/www-community/Improper_Error_Handling **Exploitability rational:** An attacker should have access to the application. **Impact rational:** By leveraging the verbose error an attacker can gain more information about the target which help in fine tuning his/her future attack.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/bus/Comps/Hudson/CareOrganization/PatientPersonalDataExports/4/
Recommendation	The application should return customized generic error messages to the user's browser. If details about the error are needed for debugging or support reasons a unique identifier may be created and displayed to the user along with the generic error message for reference. This same unique identifier can be included with the error that is logged to the server so that it can be easily correlated with the issue. References: • https://cheatsheetseries.owasp.org/cheatsheets/Error_Handling_Cheat_Sheet.html • Improper-Error-Handling-Fix-In-JAVA • Improper-Error-Handling-Fix-In-ASP.NET-Core

	• Improper-Error-Handling-Fix-In-SpringBoot
Status	**Open**

Steps to Reproduce:

1. Access the URL mentioned in the affected URL section.
2. Observe that application discloses sensitive error messages.

Supportive evidence:



8.7 WebApp: Username Enumeration

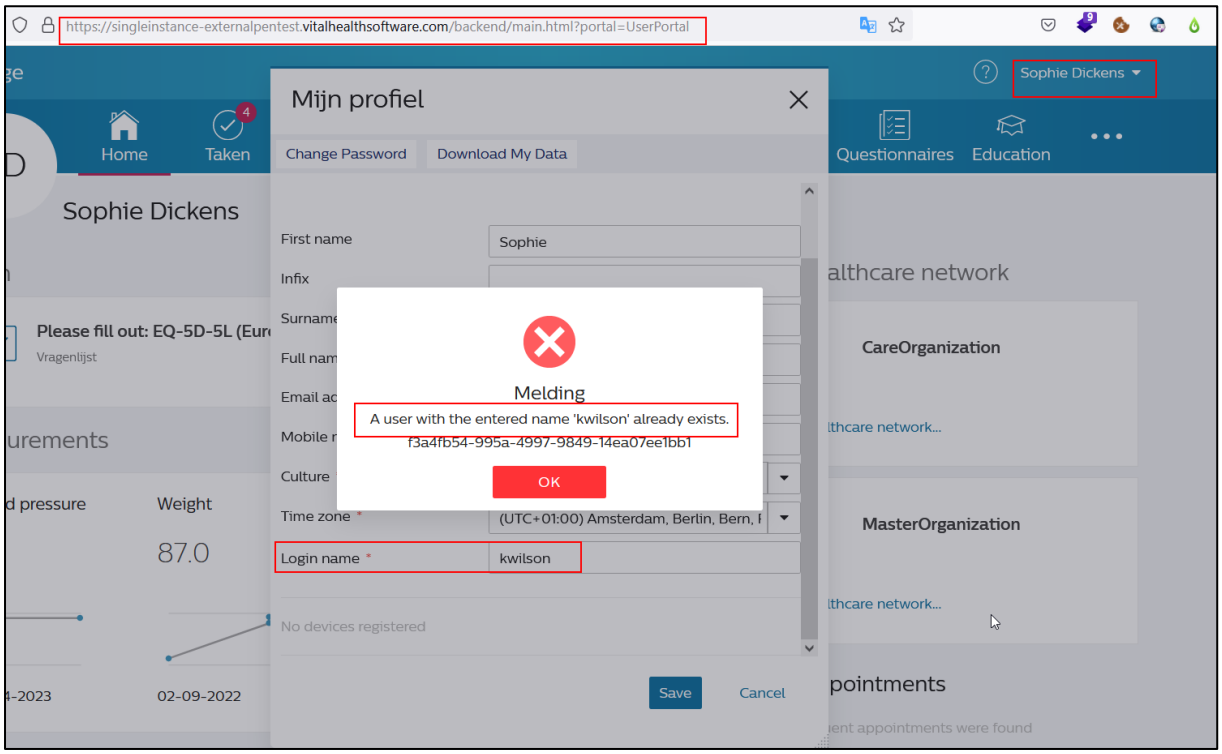
Vulnerability Title	Username Enumeration
Vulnerability Category	A5 – Security Misconfiguration
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.7 CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:U/C:L/I:L/A:N
Description	**Vulnerability Description:** During the security assessment of the product, it is observed that an attacker can find valid user of an application by using user enumeration attack. An Attacker should interact with the authentication mechanism of the application to understand if sending requests causes the application to answer in different manners. This issue exists because the information released from web application or web server when we provide a valid username is different than when we use an invalid one. **Exploitability Rationale:** An attacker uses messages in response from server while using the forget password functionality, and then he observes the valid user. **Impact Rational:** Attacker can get valid user by using this attack and it is easy for him to perform brute force attack for password. An attacker can use exposed passwords to impersonate victims in the application to steal the victim's identity or gain unauthorized access to their accounts. Reference: https://www.owasp.org/index.php/Testing_for_User_Enumeration_and_Guessable_User_Account_(OWASP-AT-002)
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/backend/main.html?portal=UserPortal
Recommendation	The server should not throw an error message, which is helpful for an end user to identify the existing user. You can do it by sending a generic error message.

	https://www.owasp.org/index.php/Testing_for_User_Enumeration_and_Guessable_User_Account_(OWASP-AT-002)
Status	**Open**

Steps to Reproduce:

1. Login to the application using patient user.
2. Go to profile and observe there is an option to change the login username.
3. It is observed that the application backend response differently for both the instances. This allows an attacker to enumerate the existing username with the system by running an intruder attack.
4. Note that the attack can be utilised to identify multiple existing uses within the system.

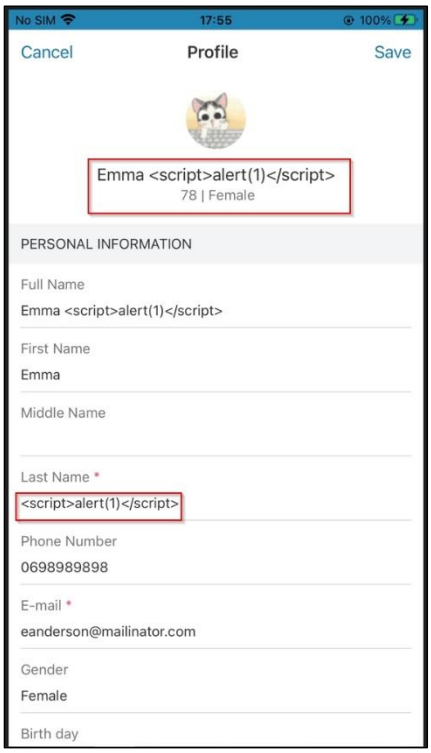
Supportive evidence:

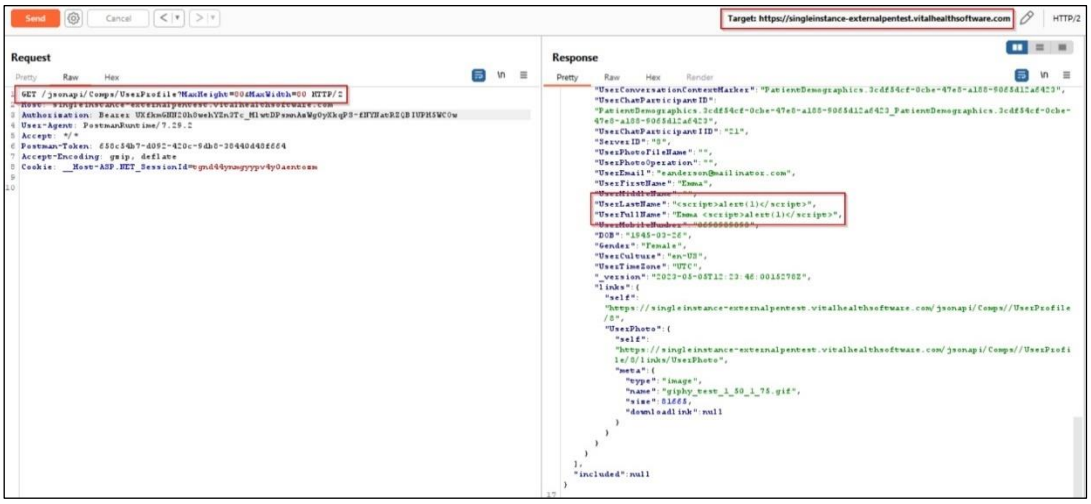


8.8 MobileApp - Both: Input Returned in Response

Vulnerability Category	M1-Improper Platform Usage
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.8 CVSS:3.0/AV:A/AC:H/PR:H/UI:R/S:U/C:L/I:L/A:L
Description	**Vulnerability Description:** When the payload is inserted as plain text in tags, the input is stored in server and echoed unmodified in the server response. This may lead to inject arbitrary JavaScript into the application. There are several instances over the application. **Exploitability Rationale:** An attacker can use the vulnerability to construct a request that, if issued by another application user, can cause JavaScript code supplied by the attacker to execute within the user's browser in the context of that user's session with the application. The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes. **Impact Rationale:** Anyone can steal cookie and can change it using stored cross site scripting.
Affected Systems/IP Address/URL	Engage-2.5.3-ci316015-prod-signed.apk Engage-2.5.3-ci316015-adhoc-signed.ipa
Recommendation	We recommend the following: • Validate the input strictly on its arrival, given the kind of content that it is expected to contain. • User input should be HTML-encoded at any point where it is copied into application responses.
Status	**Open**

Steps to Reproduce:





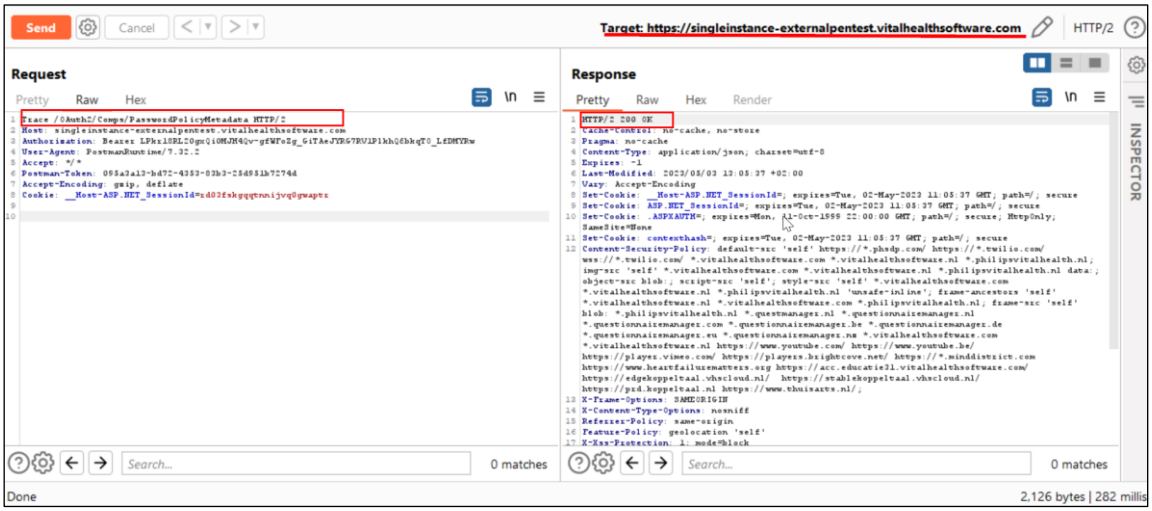
8.9 Webservices(mobile): HTTP TRACE Method Enabled

Vulnerability Title	HTTP TRACE Method Enabled
Vulnerability Category	A5 - Security Misconfiguration
Severity	**Low**
CVSS V3 Calculation	CVSS Base Score: 3.1 CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:N/A:N
Description	**Vulnerability Description:** During the security assessment of the API call, it is observed that the HTTP TRACE method is enabled on the web server. **Exploitability Rational:** The HTTP TRACE method instructs the web server to echo the entire contents of the received message back to the calling client, usually for debugging purposes. **Impact Rational:** The TRACE HTTP method can be used in conjunction with other vulnerabilities (such as cross-site scripting) to return the entire contents of an HTTP message (including server response HTTP headers) to an attacker. Since the server echoes both the request body and HTTP headers, an attacker can obtain the response to the TRACE request and can gain access to sensitive information passed via HTTP headers, including session identifiers passed via authorization header. The attacker can use this information to impersonate the victim in the application.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/OAuth2/Comps/PasswordPolicyMetadata
Recommendation	Disable the HTTP TRACE method if not required for the web server to function properly.
Status	**Open**

Steps to Reproduce:

1. Configure postman to use a proxy tool such as Burp Suite.
2. Capture and modify the request method to TRACE and then and then click on the “Go” button.
3. Observe the application response in Burp Repeater.
4. Note that the response contains the complete request, which proves TRACE method is enabled on the server.

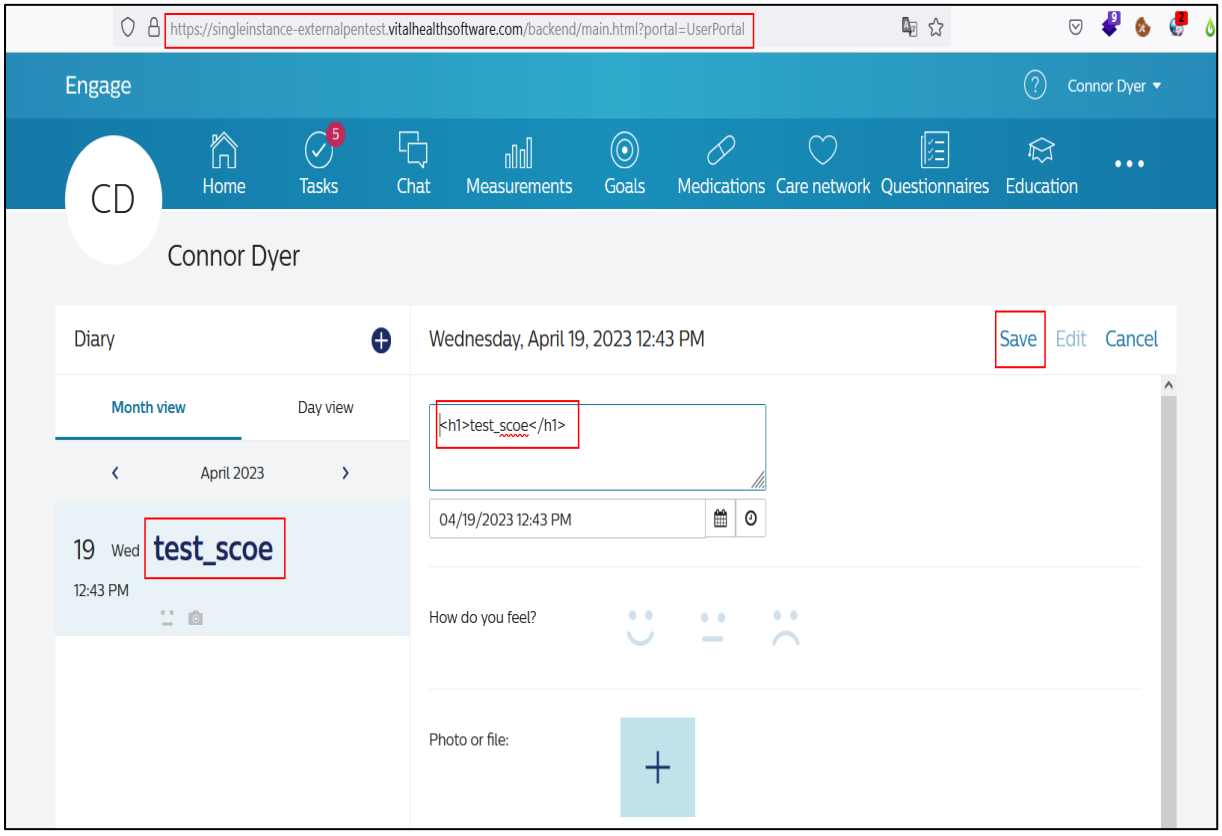
Supportive Evidence:



Informational 1 - Webapp: Stored HTML Injection

Vulnerability Title	Stored HTML Injection
Vulnerability Category	A3- Injection
Severity	**Informational**
CVSS V3 Calculation	NA
Description	**Vulnerability Description:** It is possible to inject arbitrary HTML scripts like anchor tags which references to any external websites, in the input field, which can later be triggered by another authenticated user to take them to the maliciously crafted target. **Exploitability Rational:** Any user who can login to patient/provider portal can inject the html tags. **Impact Rational:** The attacker's injected HTML is rendered and presented to the user asking for the user to redirect or enter credentials.
Affected Systems/IP Address/URL	https://singleinstance-externalpentest.vitalhealthsoftware.com/backend/main.html?portal=UserPortal
Recommendation	Implement strong input validation and filter the metacharacters from the user input.
Status	**Open**

Supportive evidence



Note: There are multiple instances of the same issue.

Informational 2 - MobileApp - Android: Insecure Local Storage

Vulnerability Category	M2-Insecure Data Storage
Severity	**Informational**
CVSS V3 Calculation	NA
Description	**Vulnerability Description:** During the security assessment, it is found that the application is insecurely storing sensitive information like JWT token in application data directory of the pronto forms mobile application. Even after logout the sensitive information are saved in local storage. Note: Giving this vulnerability as an informational finding as there is root detection enabled in the server side. **Exploitability Rational:** Attacker needs to have physical access to rooted/jailbroken devices or there should be a malware app running in background which can read through unencrypted sensitive data saved by app. **Impact Rational:** Insecure data storage can result in data loss. In the event that an adversary physically attains the mobile device, the adversary hooks up the mobile device to a computer with freely available software. These tools allow the adversary to see all third party application directories that often contain stored personally identifiable information (PII) or other sensitive information assets.
Affected Systems/IP Address/URL	Engage-2.5.3-ci316015-prod-signed.apk
Recommendation	It is recommended to follow the below instructions to secure the data. • Do not store any sensitive information in application directory. • For local storage the enterprise android device administration API can be used to force encryption to local file-stores using “setStorageEncryption”. • Ensure any shared preferences properties are NOT MODE_WORLD_READABLE unless explicitly required for information sharing between apps. • Use Android key store for any kind of key management.

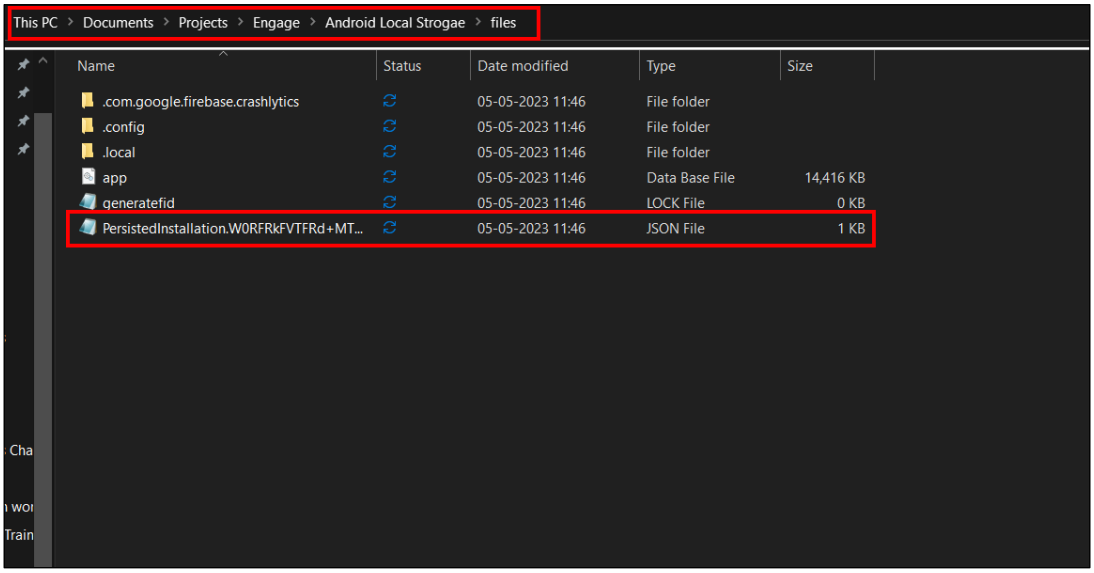
	Reference: https://owasp.org/www-project-mobile-top-10/2016-risks/m2-insecure-data-storage
Status	**Open**

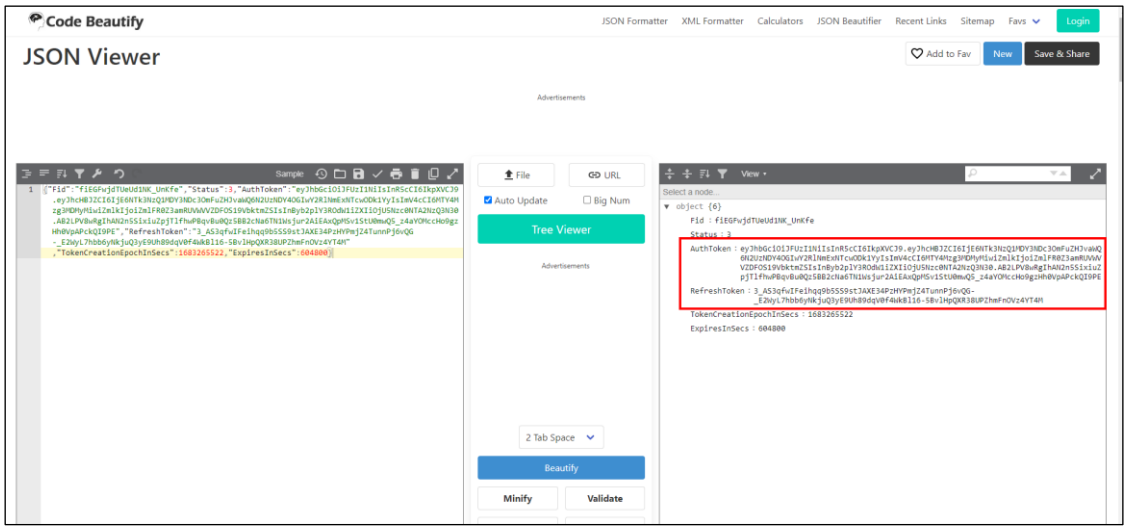
Steps to Reproduce:

1. Connect the Philips + mobile application with WinSCP and go the location

Android:"/data/data/<application package name>"

2. In the application data directory of the mobile application you find the JWT token is stored even after the user logout from application.





8. Tools Used

Scope	Tools Used
Web Application and Web services Security	Burp Suite Professional, Postman
Mobile application	Burp Suite Professional, APK tool, jd-gui, Jadx, 3utools, Objection, Frida, wireshark, sqlite-db

9. Automated Tool Report

Engage MobSF.pdf

nmap.txt

10. Manual Test Reports and Test Case Execution

2630_Engage-6.5.3_SecurityAssessmentRep

2630 Engage iOS Test Cases.xlsx

2630 Engage Android Test Cases.xls