

APPLICATION SECURITY ASSESSMENT

Technical report

Date: 11/07/2020

Client: [REDACTED]

Contact: [REDACTED]

Vulnerability Assessment - Technical Report	
Initial report date	11/07/2020 (Do not change)
Updated report date	04/02/2021
Application name	[REDACTED]
Assessment team	[REDACTED]
Assessment dates	03/07/2020 - 11/07/2020
Report approver	[REDACTED]
Remediation contact	[REDACTED]
Report shared with	[REDACTED]
Client contact	[REDACTED]
Client technical contact	[REDACTED]



1. Quick assessment summary chart

Priority	P1	P2	P3	P4	P5
Reported Findings	2	7	6	2	1
Current Findings	0	1	1	4	1
Remediation days	4	45	90	180	360

2. Scope/Synopsis of the application

The application allows the parents-to-be or new parents to explore various tools and guides to help them deal with managing the various stages of their parental leave - Pre-Leave, During-Leave and Post-Leave.

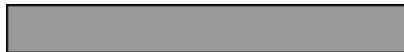
Sample client (demo):



Registered users that can be used for testing:



Sample client 2 (demo)



Registered users that can be used for testing:



3. Assessment approach

The web application assessment includes the following areas:

- Injection
- Authentication
- Session Management
- Cross Site Scripting (XSS)
- Insecure Direct Object References
- Sensitive Data Exposure
- Access control
- Cross-Site Request Forgery (CSRF)






- Unvalidated Redirects and Forwards
- Input validation
- Cryptography




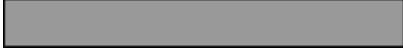
Since there is a limited window for test, every instance of a specific finding might not be uncovered. Eg if the assessor discovers SQL injection in a specific section, it may not uncover all the sections which are affected by it.

4. Assessment findings and recommendations




The following vulnerabilities were found

No	Type	Issue	Vulnerability	Recommendations
260	P1	Possible Cross Site Scripting (CLOSED)	<p>After inserting some payloads in the "First name" and saving the details application pop up showed additional html button in the confirmation pop up. This might be caused due to XSS injection.</p> <p>Affected Url: </p> <p>Affected Parameter: firstName</p> <p>Refer Figure 260.</p>	<p>Application should use proper input sanitization methods and escape all user inputs to prevent XSS vulnerabilities.</p> <p>https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet</p> <p>Update 08/08/2018: Retested. The issue is fixed. Application has implemented proper input sanitization methods and escaped all user inputs to prevent XSS injections.</p>
<p>Figure: 260</p> 				
261	P1	Request are not validated for Team member edit/delete functionality (CLOSED)	<p>It is possible to access/edit/delete team members of a different manager. Pentesters were able to modify following request and gain access to team member data of a different manager.</p> <p>Affected URL: </p>	<p>Application should first check if the user requesting the data has access to the data or not. Granting access to the data without checking the access can help attacker fetch data of any other users.</p> <p>Update 08/08/2018: Retested. The issue is fixed. Application is</p>



			<p>Affected Parameter: planId</p> <p>Refer Figure 261.</p>	<p>performing authorization by checking if the parameters that hold the plan ID / team member ID are authorized to see if they are related to the the logged in manager.</p>
<p>Figure: 261</p> 				
262	P2	<p>Missing server side validation for add/edit team member (CLOSED)</p>	<p>No server side validation is implemented for add/edit team member section. In below example the email address, gender and leave was not validated.</p> <p>Refer Figure 262.</p>	<p>Application shouldn't completely rely on client side validation of mandatory fields and should enforce server side validation for the same.</p> <p>Update 04/01/2019: Retested. The issue is fixed. Application performs server side validation for mandatory fields, validates the date selected by the user and does not rely on client side validation.</p>
<p>Figure: 262</p> 				
263	P4 (Previously P2)	<p>Possible Code Injection (Issue downgraded)</p>	<p>Application is showing an 500 error for "My Plan" section. This may be caused due to a payload used during the pentesting.</p> <p>Affected Url: </p> <p>Affected User: </p> <p>Refer Figure 263.</p>	<p>Developers will have to debug the code and identify the issue and inform the pentesters.</p> <p>Update 02/01/2019: Developers have accepted the risk and requested to downgrade the issue with following reason. There was a memory size allocation error thrown here. Due to the Payload used during the tests, the plan had over 2000 tasks. Smarty Template library encountered the memory size limitation when looping through the tasks in order to display them for the plan.</p>






				<p>In a real-life scenario, this case will never happen. A plan can consist of max 52 weeks and considering say max 4 - 5 tasks per week this can be around 300 or so.</p> <p>We can in future add pagination to this area so that only limited records are loaded.</p>
<p>Figure: 263</p> 				
264	P2	No Server side validation for Terms and condition (CLOSED)	<p>There is no server side check done to see if the "Terms and Condition" are accepted by user or not. Pentester could remove the "Terms and Condition" checkboxes and proceed with registration</p> <p>Affected Url:</p>  <p>Refer Figure 264.</p>	<p>Application shouldn't completely rely on client side validation of mandatory fields and should enforce server side validation for the same.</p> <p>Update 31/12/2018: Retested. The issue is fixed. Application performs server side validation to check if Terms and Conditions checkbox field value is ticked or not instead of relying only on client side validation.</p>
<p>Figure: 264</p> 				
265	P2	No security question for password reset	<p>Application doesn't ask for security questions while changing the password.</p>	<p>In order to verify the genuinity of the user while changing the password, application should register 6 security questions in advance from the user and randomly ask user 3 security questions at password reset.</p>
266	P4 (Previously P2)	No Email verification done on registration and my account page (Issue downgrade)	<p>Application does not verify email address during registration. Also while changing email address in my account page, the email is saved without verification.</p>	<p>Application should verify the email address by sending and confirmation email to the user and asking to click on the verification link from the email.</p> <p>Update 02/01/2019: Developers have accepted the</p>




		d)		<p>risk and requested to downgrade the issue with following reason.</p> <p>The user if verified by 2 methods:</p> <p>a) Email domain verification - wherein the user receives an email to verify themselves by clicking a link in the email.</p> <p>b) Verification against CSV fields - wherein the client sends the eligibility list of their employees and when users register they are verified against the fields that the company decides (like employee number, surname or email)</p>
273	P2	Possible code injection (CLOSED)	<p>Following account might be possibly affect with some payloads used during pentesting. Deletion of tools is not working in the account. While adding the tools the pentesters were trying to manipulate and pass payloads in the parameter.</p> <div style="background-color: #cccccc; width: 150px; height: 15px; margin: 5px 0;"></div>	<p>Developers will have to debug and identify the issue.</p> <p>Update 14/08/2018: Retested. The issue is fixed. The payloads were getting inserted into the database where the actual ID of the tools should have got inserted. Due to this, the deletion action was not working as it threw a Javascript error. Parameters are validated and sanitized to allow only valid ID's while inserting tools to "My Folder" area.</p>
277	P2	No Server side validation done on registration form (CLOSED)	<p>Pententer could register a user without a First name though the First name field was mandatory.</p> <p>Affected URL:</p> <div style="background-color: #cccccc; width: 150px; height: 15px; margin: 5px 0;"></div> <p>Refer Figure 277.</p>	<p>Application shouldn't completely rely on client side validation of mandatory fields and should enforce server side validation for the same.</p> <p>Update 02/01/2019: Retested. Issue is fixed. Application performs server side validation for the mandatory field and doesn't rely only on client side validation for mandatory field. During the retest the attempt was made to bypass First name</p>




				field but the application detected the missing field.
Figure: 277 				
267	P3	Allows to configure error message on registration page (CLOSED)	<p>It is observed that the hidden field verFields can be manipulated to show custom error message.</p> <p>Since the form post values back to the application it could be that this values are used to perform checks</p> <p>Please check other hidden fields in the application as well.</p> <p>Refer Figure 267.</p>	<p>Error messages shown in the application should be shown from server side and should be configurable from the client side using hidden parameters.</p> <p>Update 29/01/2019: Retested. The issue is fixed. Application was relying on hidden field "verFields" for displaying error messages. Application has stopped the use of hidden field and currently relies on server side validation to show the error messages.</p>
Figure: 267 				
268	P3	Allows to configure error message on my account page (CLOSED)	<p>Below is formData field's POST value for which we have changed value of verFields</p> <pre>{ "formId": "myAccountForm", "hEmpFields": "empNo, ", "hEmpSave": "empNo, ", "hEmpValues": "", "verFields": "Test TEST", "firstName": "Jones", "lastName": "Byrne", "mobile": "07777 7777", "email": "</pre>  <pre> ", "altEmail": "", "oldPassword": "", "password": "", "confirmPassword": "", "empNo": "dddd", "veriSave": "" }</pre> <p>formData.verFields can be manipulated to show custom error message.</p>	<p>Error messages shown in the application should be shown from server side and should not be configurable from the client side using hidden parameters.</p> <p>Update 29/01/2019: Retested. The issue is fixed. Application was relying on field "verFields" for displaying error messages. Application has stopped the use of this field and currently relies on server side validation to show the error messages.</p>



			Refer Figure 268.	
Figure: 268 				
269	P3	Session id not regenerated (CLOSED)	Application does not regenerate session id after the auto login which happens after registration. After clicking the "Get Started" button on the successful registration page, the application logs in the user automatically but does not change the session id. This can help attacker to execute a session fixation attack.	Session Id should always be regenerated in the application during important stages like user login, password change or security questions change and password reset. Update 02/01/2019: Retested. The issue is fixed. Session ID has been regenerated at all strategic points like the user login, user auto login on registration and password change.
270	P3	Strict transport security not enforced (CLOSED)	The application fails to prevent users from connecting to it over unencrypted connections. An attacker able to modify a legitimate user's network traffic could bypass the application's use of SSL/TLS encryption, and use the application as a platform for attacks against its users. This attack is performed by rewriting HTTPS links as HTTP, so that if a targeted user follows a link to the site from an HTTP page, their browser never attempts to use an encrypted connection. The sslstrip tool automates this process.	Enable HTTP Strict Transport Security (HSTS) by adding a response header with the name 'Strict-Transport-Security' and the value 'max-age=expireTime', where expireTime is the time in seconds that browsers should remember that the site should only be accessed using HTTPS. Consider adding the 'includeSubDomains' flag if appropriate. Reference: https://www.owasp.org/index.php/HTTP_Strict_Transport_Security_Cheat_Sheet Update 28/12/2018: Retested. The issue is fixed. Application implements Strict transport security by adding the "Strict-Transport-Security" response header to the web application responses.
271	P3	Frameable	No appropriate X-Frame-Options	To effectively prevent framing



		Response (potential clickjacking) (CLOSED)	or Content-Security-Policy HTTP header. It might be possible for a page controlled by an attacker to load it within an iframe. This may enable a clickjacking attack, in which the attacker's page overlays the target application's interface with a different interface provided by the attacker.	attacks, the application should set X-Frame-Options header with value DENY to prevent framing altogether, or the value SAMEORIGIN to allow framing only by pages on the same origin as the response itself. Example: X-frame-options: SAMEORIGIN https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet Update 31/12/2018: Retested. The issues is fixed. Application implements x-frame-options: "SAMEORIGIN" in header.
272	P3	Using components with known vulnerabilities	Outdated javascript libraries used which has vulnerabilities. Refer Figure 272.	While no such exploitable vulnerabilities were observed within the current application version, it is strongly recommended to update this component to prevent future manifestations as the application evolves.
Figure: 272 				
274	P4	Information disclosure	Application using PHPSESSID as the session id name. This can reveal to the attacker that this is a session id and its using a php as a programming technology.	Ideally use a session id name which is less revealing.
275	P4	Web Browser XSS Protection is not enabled	Web Browser XSS Protection is not enabled, or is disabled by the configuration of the 'X-XSS-Protection' HTTP response header on the web server.	The x-xss-protection header is designed to enable the cross-site scripting (XSS) filter built into modern web browsers (Internet Explorer 8+, Chrome, Mozilla and Safari) This is usually enabled by default, but using it will enforce it. The recommended



				<p>configuration is to set this header to</p> <pre>X-XSS-Protection: 1; mode=block</pre> <p>PHP:</p> <pre>header("X-XSS-Protection: 1; mode=block");</pre> <pre>.htaccess <IfModule mod_headers.c> Header set X-XSS-Protection "1; mode=block" </IfModule></pre> <p>This header will enable XSS filtering. Rather than sanitizing the page, the browser will prevent rendering of the page if an attack is detected.</p>
276	P5	X-Content-Type-Options header is not set	<p>The X-Content-Type-Options header is not set This could allow the user agent to render the content of the site in a different fashion to the MIME type.</p>	<p>Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.</p>



5. Vulnerability threat category

Rating	Definition	Impact
Priority 1 (P1)	Issues that pose a clear and present danger to the confidentiality, availability and integrity of the system or data. Any existing mitigating controls are ineffective or insufficient. Includes readily exploitable issues that pose severe financial, brand image, or regulatory impact if discovered/exploited. Any issue that poses a direct and probable threat to the company's confidential information or customer NPI falls into this category.	For PRIORITY 1 vulnerabilities modifications, e.g. security patches, fixes, etc. MUST be deployed within 4 calendar days from date of Notification. Needs to be re-tested. Confirmation on deployment on live server needed.
Priority 2 (P2)	Issues that pose the highest and/or significant immediate risk to the confidentiality, availability and integrity of the system or data. Any existing mitigating controls are ineffective or insufficient. Includes readily exploitable issues that pose significant financial, brand image, or regulatory impact if discovered/exploited. Any issue that allows compromise of the infrastructure or allows anonymous access to authenticated systems fall into this category. Any issue that has a high probability of occurrence.	For PRIORITY 2 vulnerabilities modifications, e.g. security patches, fixes, etc. MUST be deployed within 45 calendar days from date of notification. Needs to be re-tested. Confirmation on deployment on live server needed.
Priority 3 (P3)	Issues that pose a moderate risk to the confidentiality, availability and integrity of the system or data, and mitigating controls that are either nonexistent or ineffective. Includes readily exploitable issues that pose moderate business impact if discovered/exploited.	For PRIORITY 3 vulnerabilities modifications, e.g. security patches, fixes, etc. MUST be deployed within 100 calendar days from date of notification. Needs to be re-tested. Confirmation on deployment on live server needed.
Priority 4 (P4)	Issues that pose a low risk to the confidentiality, availability, and	For PRIORITY 4 vulnerabilities it is recommended that modifications,



	<p>integrity of the system or data, but could make the application less safe or introduce a problem in the future. This includes potentially dangerous issues that are not directly exploitable. Includes readily exploitable issues that pose low business impact if discovered/exploited. Any issues that have existing effective mitigating controls.</p>	<p>e.g. security patches, fixes, etc. may be remediated based on system maintenance schedules.</p> <p>No retest required</p>
Priority 5 (P5)	<p>Anomalous items that either do not pose apparent security risks to the confidentiality, availability and integrity of the Systems or data. These computing system vulnerabilities rated Priority 5 are provided for general improvement suggestions or simply to share information which may be of interest to the technology owners. Priority 5 advisories are not tracked or notified.</p>	<p>For Priority 5 vulnerabilities remediation is not required or monitored.</p> <p>No retest required</p>

