Manual for operation

Version: 2024.0-SNAPSHOT.33 20.01.2025 9:39:09

Contents

1. Safety tests

The security of the WebJET CMS is dependent on its correct configuration and correct access rights settings. The information on this page must be set up prior to going into production operation and then checked at least quarterly and always prior to performing security tests.

1.1. Setting up the system

1.1.1. Groups of rights

It is also possible to modify program files via WebJET CMS, so it is necessary to set permission restrictions before penetration testing. We recommend to create the following groups of rights (https://docs.webjetcms.sk/latest/en/admin/users/perm-groups). **Other user groups, or users directly, should not be allowed the below unit rights**.

User management

The group contains permissions that allow you to modify permissions. A user with such a group of rights must be cautious enough to know that he has the highest privilege options (because he can set rights for himself or for other users).

A user with this privilege can compromise the entire system (e.g., he can set permissions so that he can delete all web pages or all files).

Set the following rights for the rights group:

- administrators management - the right allows to set permissions for users in the administration
- Rights groups - the right allows you to set permissions on groups

Programmer

The default setting should be that users (editors) cannot upload and modify program files. However, the programmer often needs to make a quick change (`hotfix`) of the program code and thus needs to modify the program files as well. At the same time, it has added rights to modify all configuration variables and edit all translation texts.

A user with this privilege can compromise the entire system (e.g., upload malicious code that can perform any operation on the server, including deleting files on the server or completely wiping the database).

- Unlimited file uploads (extensions and sizes)
- Configuration - view all variables
- Text editing - view all texts

Note: the list of conf. variables without the "Configuration - display all variables" permission is set in the conf. variable `configEnabledKeys` , list of translation keys without the "Text editing - display all texts" permission in the conf. variable `propertiesEnabledKeys` . HTML code is also filtered in the translation keys, the rules are described below in the section `Stored XSS cez úpravu prekladových kľúčov` .

In addition to permissions, you must allow write access to the file system directories:

- `/apps` - contains application code
- `/components` - contains application code
- `/templates` - includes design templates

If the environment is deployed directly from the GIT repository and you do not expect to execute `hot-fixov` directly via WebJET CMS you do not need to set the above rights for writing to the file system. Additionally, for this case, we recommend setting write permissions on the file system for directories only (other directories and files have read-only permissions):

- `/images` - contains images uploaded by CMS editors
- `/files` - contains files uploaded by CMS editors
- `/shared` - contains images and files uploaded by CMS editors and shared between domains
- `/WEB-INF/tmp` - contains temporary CMS files

- `/WEB-INF/imgcache` - contains generated image thumbnails and cutouts for use via `/thumb` prefix
- `/WEB-INF/formfiles` - contains files uploaded via forms on the website created via the Forms application

1.1.2. Configuration

Set and check the following configuration variables (in Settings->Configuration administration):

- `defaultDisableUpload=true` - activates a mode in which the user has file system rights only for the configured directories. If no directories are set he will not have write permissions to any directory.
- `emailProtectionSenderEmail` - set a suitable email address type `noreply@domena.sk`, which is used as the email address of the sent emails (the original value is set to `Reply-To` email headers).
- `adminEnabledIPs` - contains a comma-separated list of IP addresses from which access to `/admin` parts.
- `multidomainAdminHost` - allows you to set a separate domain address for access to `/admin` parts, e.g. `cms.domena.sk`. After setting up, the call will be `/admin` addresses on other domains return a 404 error - Page does not exist.
- `serverBeyoundProxy` - if the application server is behind a Load Balancer / proxy you need to set the value to `true` (otherwise set the value `false`). The Load Balancer must then send the following in the HTTP header `x-forwarded-for` the IP address of the visitor to the website and in the header `x-forwarded-proto` protocol (`http` or `https`). Verify in an audit (e.g. after filling in a form on the website) that the IP address of the website visitor is correctly recorded.
- `serverName` - default `unknown` - sets the HTTP header value `Server` for HTTP response. If you have an application server behind the Load Balancer/proxy, verify the value of this header in the HTTP response and possibly set it to a suitable unknown value on the Load Balancer/proxy.

Restrictions for uploaded files by editors in the administration:

- `FCKConfig.UploadMaxSize[Default][image]` - default 0 - size limit in kB for uploading **Images**, we recommend to set it to 10000 for uploading max 10 MB image
- `FCKConfig.UploadMaxSize[Basic][image]` - default 2048 - size limit in kB for uploading **Images** for users who **do not have the right Complete menu in the editor**
- `FCKConfig.UploadMaxSize[Default][file]` - default 0 - size limit in kB for uploading **files**, we recommend to set it to 50000 for uploading max 50 MB file
- `FCKConfig.UploadMaxSize[Basic][file]` - default 2048 - size limit in kB for uploading **files** for users who **do not have the right Complete menu in the editor**
- `FCKConfig.UploadFileTypes[Default][image]` - default empty = no limits - type limits **Images**, we recommend setting to `jpg,jpeg,png,gif,svg,mp3,mp4`. The possibility of allowing the SVG suffix needs to be considered, see potential risk below in the block `Stored XSS cez SVG obrázok`.
- `FCKConfig.UploadFileTypes[Basic][image]` - default `jpg,jpeg,png,gif,mp4` - type limits **Images** for users who **do not have the right Complete menu in the editor**
- `FCKConfig.UploadFileTypes[Default][file]` - default empty = no limits - type limits **files**, we recommend setting to `pdf,docx,xlsx,pptx,ppsx,zip,rtf`
- `FCKConfig.UploadFileTypes[Basic][file]` - default `doc,docx,xls,xlsx,pdf,zip,rtf` - type limits **files** for users who **do not have the right Complete menu in the editor**

You can also check the following conf. variables:

- `overviewJsonUrl` - defines the URL from which the list of news in WebJET is read. If users do not have internet access from a browser, you can set this to `/admin/v9/json/` for reading from a local instance. But users will not see the news in new versions of WebJET CMS.
- `springSecurityAllowedAuths` - list of allowed authorization methods for REST services, by default `basic,api-token`. Set to blank if the project does not need other than a standard form login. You must restart the application server after changing the value.

1.1.3. HTTP headers

In the Configuration application, you can set the values of the security headers:

- `contentSecurityPolicy` - header setting `Content-Security-Policy` . Restrictions on how the page should load different resources. If you have an httpS certificate you can set it to:
  `default-src 'none'; script-src https: blob: data: 'unsafe-inline' 'unsafe-eval'; worker-src https: blob:; child-src https`
  Blank by default (header is not set).
- `contentSecurityPolicySvg` - specific limitation for SVG images due to their different processing in Internet Explorer.
- `featurePolicyHeader` - HTTP header value `Feature-Policy/Permissions-Policy` (e.g.:
  `microphone 'none'; geolocation 'none'`), more at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Feature_Policy (https://developer.mozilla.org/en-US/docs/Web/HTTP/Feature_Policy). Empty by default.
- `refererPolicy` - HTTP header settings `Referrer-Policy` , we recommend setting to `same-origin` . Default `same-origin` .
- `serverName` - header value `Server` in HTTP responses. Default `unknown` . It is not possible to set it to an empty value, because then the application server puts the header there.
- `strictTransportSecurity` - empty by default - sets the HTTP header Strict-Transport-Security (https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security) in HTTP responses, which ensures HTTP requests are redirected to secure httpS, we recommend setting to: `max-age=31536000 ; includeSubDomains` . It requires that the application server is accessible via the secure httpS protocol.
- `xContentTypeOptions` - header value `X-Content-Type-Options` to set the determination of file types by content (ignoring suffixes). Default `nosniff` .
- `xFrameOptions` - header value `X-Frame-Options` for protection against CSRF attack. Default `SAMEORIGIN` .
- `xRobotsTagValue` - header value `X-Robots-Tag` for URLs set in the variable `xRobotsTagUrls` . Default `noindex, nofollow` .
- `xRobotsTagUrls` - comma-separated list of URL beginnings for setting the header `X-Robots-Tag` . If the list contains a value `NOT_SEARCHABLE_PAGE` the header is set even for pages that have search disabled. Default `/components/,NOT_SEARCHABLE_PAGE` .
- `xXssProtection` - header value `X-XSS-Protection` for protection against XSS attack. Default `1; mode=block` .

Setting the header `Access-Control` to access REST services from other servers:

- `accessControlAllowOriginValue` - header value `Access-Control-Allow-Origin` for URL set in variable `accessControlAllowOriginUrls` . You can use a macro in the value (see below). By default set to `{HTTP_PROTOCOL}://{SERVER_NAME}:{HTTP_PORT}` .
- `accessControlAllowOriginUrls` - comma-separated list of URL beginnings for setting the header `Access-Control-Allow-Origin` . Default set to `/rest/,/private/rest/,/admin/rest/` .
- `accessControlAllowHeaders` - header setting value `Access-Control-Allow-Headers` , is set only when generating the header `Access-Control-Allow-Origin` . Default
  `Origin, Accept, X-Requested-With, Content-Type, Access-Control-Request-Method, Access-Control-Request-Headers, x-csrf-tok`
- `accessControlAllowMethods` - header setting value `Access-Control-Allow-Methods` , is set only when generating the header `Access-Control-Allow-Origin` . Default `HEAD,POST,GET,OPTIONS,PUT` .
- `accessControlMaxAge` - header setting value `Access-Control-Max-Age` , is set only when generating the header `Access-Control-Allow-Origin` . Default `1800` .
- `accessControlAllowedOrigins` - if not empty, requires a header in the request `origin` whose value must be in this list (comma/newline separated list). It is set only when generating the header `Access-Control-Allow-Origin` . Blank by default.

If you need to set a different HTTP header you can use the application HTTP headers (https://docs.webjetcms.sk/latest/en/admin/settings/response-header/README) in the Settings section.

You can use a macro in the value `{HTTP_PROTOCOL}, {SERVER_NAME}/{DOMAIN_NAME}/{DOMAIN_ALIAS}, {HTTP_PORT}` , which will be replaced by the value obtained on the server. `SERVER_NAME` is a domain name from `request.getServerName()` , `DOMAIN_NAME` a `DOMAIN_ALIAS` are the domain or alias values set in the website. The value of `{INSTALL_NAME}` represents the name of the installation. Value `{HEADER_ORIGIN}` contains the HTTP header value `origin` .

In older installations it was also possible to set HTTP headers via conf. variable `responseHeaders` in which you can set the header for the URL prefix (the beginning of the URL). On each line you enter a value in the format: `url-prefix:hlavička:hodnota` , for example:

```
/admin:X-Accel-Buffering:no
/rest/calculators/:Access-Control-Allow-Origin:*
/rest/calculators/:Access-Control-Allow-Headers:origin,x-requested-with,access-control-request-headers,content-
type,access-control-request-method,accept,x-csrf-token
/rest/calculators/:Access-Control-Allow-Methods:GET,OPTIONS
```

Values set via conf. variable `responseHeaders` are global regardless of the current domain.

1.1.4. Password rules

Password rules can be set via the following configuration variables (the default value is given in brackets next to the variable):

- `passwordAdminMinLength` - Specifies the minimum length of the password for the administrator (5).
- `passwordAdminMinCountOfSpecialSigns` - Specifies the minimum number of occurrences of special characters in the administrator password (0).
- `passwordAdminMinUpperCaseLetters` - Specifies the minimum number of occurrences of uppercase letters in an administrator password (1).
- `passwordAdminMinLowerCaseLetters` - Specifies the minimum number of occurrences of lowercase letters in the password for the administrator (0).
- `passwordAdminMinCountOfDigits` - Specifies the minimum number of occurrences of numbers in the password for the administrator (1).
- `passwordAdminExpiryDays` - Specifies the number of days the password is valid for the administrator. After the time expires, the user will be prompted to change the password. A value of 0 means that the password expiration (0) is not checked.

Similarly, password rules can be set for logging into the secure area of the website (not the administration), the variables are the same, but do not contain the expression `Admin`. So the variable name is e.g. `passwordMinUpperCaseLetters`.

By setting the configuration variable `isGoogleAuthRequiredForAdmin` at `true` will be for access to `/admin` parts required two-factor verification. Each user must set it up in advance in the administration by clicking on their name in the top right and selecting the option **Two-step verification**, or by opening the page `/admin/2factorauth.jsp`.

We recommend that you set up two-factor authentication for at least all accounts that can be used to manage user accounts and privileges and set system configuration.

WebJET checks the history when changing the password and does not allow the same password to be used again. This is affected by the following conf. variables:

- `passwordHistoryLength` - the number of used user passwords that are remembered in the history (by default 6).
- `passwordHistoryEnabled` - if set to `true` the password history is checked in the database and it is not allowed to change the password that was used in the past (by default `true`).

The following variables are used when requesting a password change:

- `passwordResetValidityInMinutes` - time validity in minutes for the sent password change link (default 30).
- `changePasswordPageUrl` - address of the password change page (default `/components/user/change_password.jsp`).

1.1.5. Blocking login

After an incorrect combination of username and password, WebJET blocks further logins from the same IP address. It is possible to set the following conf. variables:

- `logonBlockedDelay` - Time in seconds during which it will not be possible to log in again after entering the wrong name/password (default 10).
- `logonBlockedAfterUnsuccessCount` - the number of unsuccessful logins after which the delay defined in `logonLoginBlockedDelay` (default 5).
- `logonLoginBlockedDelay` - the time in seconds during which it will not be possible to log in again after entering a bad password; and `logonBlockedAfterUnsuccessCount` the number of failed logins for the specified login name (default 60).

Thus, a value of 10 seconds is applied by default (`logonBlockedDelay`), if entered more than 5 times in a row (`logonBlockedAfterUnsuccessCount`), a delay of 60 seconds is applied (`logonLoginBlockedDelay`).

During the login lockout time, the failed attempts counter is still not incremented and the time is not extended, since no login code is called at all.

1.1.6. Password hashing algorithm

From version `2022.40` the BCrypt algorithm is used to hash passwords in the implementation `org.springframework.security.crypto.bcrypt.BCrypt` .

Possible settings:

- `bcryptSaltRounds` (default 12) - log2 number of flipping repetitions
- `passwordHashAlgorithm` (default `bcrypt`) - name of the hashing algorithm, possible values `bcrypt` or `sha-512` .

Earlier versions used the algorithm `SHA-512` with 100 repetitions. Legacy password hash values are changed to the bcrypt algorithm when the user's password is changed. To force the algorithm change you can set the conf. variable `passwordAdminExpiryDays` to a non-zero value, which will prompt the user to change the password after logging in.

1.1.7. Logging into administration

Logging into the administration is also affected by the configuration variables mentioned above:

- `adminEnabledIPs` - contains a comma-separated list of IP addresses from which access to `/admin` parts.
- `multidomainAdminHost` - allows you to set a separate domain address for access to `/admin` parts, e.g. `cms.domena.sk` . After setting up, the call will be `/admin` addresses on other domains return a 404 error - Page does not exist.
- `isGoogleAuthRequiredForAdmin` - enabling two-factor authentication when logging into the administration `/admin` .
- `clusterMyNodeType` - in the case of a cluster sets the node mode, only nodes set to `full` include administration and allow you to log in.
- `auditDontLogUsrlogon` - after setting to `true` normal (non-administrator) user logins will not be audited. Suitable for a highly loaded intranet where it unnecessarily overwhelms the audit (by default `false`).

In addition, a user who successfully authorises must meet the following criteria:

- `Schválený používateľ` - the account must have that option selected.
- `Začiatok platnosti` - if the value entered must be older than the current date.
- `Koniec platnosti` - if the value entered must be greater than the current date.
- `Povoliť vstup do admin sekcie (správa web sídla)` - account must have this option enabled, otherwise access to the administration is not possible.

In case of special requirements for user login/authentication, it is possible to implement a custom Java login class and set it via a conf. variable `adminLogonMethod` . The specified Java class is then used instead of the default login.

When logging in with incorrect details blocks login.

1.1.8. Authentication against LDAP server

The following conf. variables can be set when authenticating a user against an LDAP server:

- `ldapProviderUrl` - LDAP server URL for LDAP login in the form `ldap://ldap.local:389/DC=firma,DC=com??base` .
- `ldapPassword` - the login name of the technical user for retrieving LDAP data.
- `ldapUsername` - the login password of the technical user to retrieve LDAP data.
- `ldapUseSslProtocol` - uses SSL when communicating with the LDAP server. You must have SSL enabled on port 636 of the LDAP server. If ldapS:// is used, leave the value at false.
- `NTLMForbiddenURL` - URL of the access denied (by default `/500.jsp`).
- `NTLMDomainController` - the name of the domain controller.
- `ldapDomainAppend` - if it is necessary to log in with the whole domain, it is possible to specify its addition to the specified user login name.
- `ldapSecurityPrincipalDn` - sets a special `SECURITY_PRINCIPAL` e.g. `cn=!USERNAME!,dc=ad,dc=interway,dc=sk` with the understanding that `!USERNAME!` is replaced by the login name. If it is empty it is used `ldapUsername+ldapDomainAppend` .
- `ldapFilter` - login filter for LDAP login with which account lookup is performed (default `(&(objectClass=Person) (&(sAMAccountName=!USERNAME!)))`).
- `basicNtlmLogonAttrs` - A list of attributes to read from the LDAP server when logging in. If empty, only the login is verified and the user is not updated with the values in the LDAP server. Default `mail,title,givenName,sn,streetAddress,l,postalCode,co,company,telephoneNumber,mobile,description,memberOf,distinguishedN`
- `ntlmDefaultUserPhoto` - if set to a non-empty value and the user does not have a photo in LDAP sets the photo to the specified URL.

Rights settings:

After logging in, the user group name and group rights are automatically checked against the groups in LDAP (attribute `memberOf`). If the name matches a group in WebJET, the user is assigned. In addition, it is possible to set:

- `NTLMAdminGroupName` - The name of the group in LDAP that identifies that the account has administrative access (e.g. WebJETAdmins).
- `passwordProtectedAutoId` - A comma-separated list of user group IDs that will be assigned automatically to a user after a successful login.

1.1.9. Configuring a Tomcat Application Server

We assume that the website/application will be accessible via secure httpS protocol. Therefore, it is necessary to set the attribute `secure="true"` , [HTTP/AJP connector (https://tomcat.apache.org/tomcat-9.0-doc/config/http.html)](https://tomcat.apache.org/tomcat-9.0-doc/config/http.html) so that the session cookie is only accessible via the secure httpS protocol. You make the setting in the file `tomcat/conf/server.xml` . Attribute `useBodyEncodingForURI="true"` sets the same character encoding for the URL as is used for the page body.

```xml
<Connector
    ...
    secure="true"
    useBodyEncodingForURI="true"
    ...
/>
```

if you use an insecure HTTP protocol, the session cookie will not be accepted by the browser and the session will not be held (this will be shown by a repeated display of the login window as soon as you enter the correct credentials).

To avoid displaying the application server version when an error is displayed `Tomcat` a `Stack Trace` is required in server.xml to `<Host` element add configuration [ErrorReportValve (https://tomcat.apache.org/tomcat-9.0-doc/config/valve.html#Error_Report_Valve):](https://tomcat.apache.org/tomcat-9.0-doc/config/valve.html#Error_Report_Valve)

```xml
<Host ...>
    <Valve className="org.apache.catalina.valves.ErrorReportValve"
        showReport="false"
```

```xml
        showServerInfo="false" />
</Host>
```

if necessary you can also create a static html page in coding `utf-8` with an error message and configure it as:

```xml
                                                                    xml
<Valve className="org.apache.catalina.valves.ErrorReportValve"
    errorCode.400="webapps/error400.html"
    errorCode.0="webapps/errorOthers.html"
    showReport="false"
    showServerInfo="false" />
```

In the configuration file `server.xml` we recommend to set the option correctly `defaultHost`. Attackers can modify the header `Host` and thus direct the request from the Internet, e.g. to an administration host that might not be accessible from the Internet (if both the administration and public nodes of the cluster are running on the same application server). An example is the use of `localhost` that can be assigned to the administration server and thus the modified request can end up on the administration node.

Value `defaultHost` we recommend to direct to the non-existent `<Host>` element, in which case the application server will declare an error that such `host` he doesn't know. Thus, the application server does not handle unknown domains. The disadvantage of this solution is that after adding a new domain, it must be added as `<Alias>` also to the application server.

If you use Load Balancer you need to ensure that it only sends known (whitelist) domains to the application servers. For unknown domains it must respond with an error.

```xml
                                                                    xml
<Engine name="Catalina" defaultHost="localhost">

    <Host name="localhost"...>
        <Alias>admin.domain.eu</Alias>
    </Host>
```

1.1.10. Change notifications

We recommend setting up notifications sent to the security engineer for the following types of events:

- `CONF_UPDATE` a `CONF_DELETE` - change/delete configuration variable
- `PROP_UPDATE` a `PROP_DELETE` - changing/deleting the translation key (JavaScript code can also be inserted via the translation key)

You can set up notifications in the administration under Audit->Notifications. The security engineer will be aware of these changes and can react if an attack is suspected.

1.1.11. Server security

The security of the server itself and the software used is also important. Update the software to the latest supported versions. It is also advisable to install an antivirus on the server, which will check the uploaded files and, if a virus is detected in a file, will prevent access to that file.

1.2. Load Balancer settings

If Load Balancer is preloaded in front of the application servers, it is necessary to ensure:

- Route only defined domains to application servers so that an attack cannot occur by modifying `Host` headers. Load Balancer must not allow sending an unknown domain to the application server.
- WebJET takes the IP address setting from the HTTP header `X-Forwarded-For` when setting the configuration variable `serverBeyoundProxy=true`. So it is necessary to ensure that such HTTP header is not coming from the Internet, but Load Balancer always rewrites it to the correct value - the IP address of the visitor. The same applies to the HTTP header `x-forwarded-proto`. Incorrect header settings can lead to accessing parts that are only allowed for specific IP addresses. such as administration.

1.3. WAF settings

If the application server is predefined `Web Application Firewall/WAF` need to be set **exceptions for administration**. Some HTTP requests in the administration can be detected as an XSS/SQL Injection attack because the HTTP request can send JavaScript code or an SQL statement. An example is saving a web page where the necessary JavaScript code may be inserted in the HTML code field in the header, or saving records in the Scripts application where JavaScript code is directly inserted.

The ideal solution is to use a clustered solution with a dedicated CMS node on the local network that is inaccessible from the external environment. For this case, the WAF for the CMS node can be omitted.

The administration uses REST services starting at the URL `/admin/rest`, see recommendations for URL rules (https://docs.webjetcms.sk/latest/en/custom-apps/spring/rest-url). On WAF you need to set exceptions for URLs starting with:

- `/admin/rest/web-pages` - storage of web pages
- `/admin/rest/components/insert-script` - Application Scripts
- `/admin/v9/settings/translation-keys` - translation keys - in some cases it may be necessary to insert HTML code into the translation key
- `/admin/rest/settings/configuration` - configuration, applies similar to the translation keys
- `/admin/searchall.jsp` - search in the administration, it may be necessary to search for HTML/JavaScript expression
- `/admin/replaceall.jsp` - replacing an expression in the administration, similar to the search
- `/admin/updatedb.jsp` - execution of the specified SQL statement

Other URLs should be set based on the applications you use.

1.4. Resolving security findings

- `Sensitive Data Exposure`

The type and version of the web server used was detected through the error responses from the server.

Solution: verify HTTP header settings `Server`, in WebJET can be rebuilt in the configuration variable `serverName`, see above.

- `RCE via uploaded JSP file`

WebJET CMS allows uploading of arbitrary files, including JSP files, which allow the execution of arbitrary commands on the running server.

The error can be prevented by setting the file upload rights, or by preventing the writing of program files altogether.

Solution: edit the rights setting for file upload via configuration variables `FCKConfig.Upload*`, see above.

- `MaliciousFileUpload`

The server lacks virus scanning, so it is possible to upload malicious files to the server.

Solution: install an antivirus on the server.

- `Missing Secure cookie flag`

Session cookie (`JSESSIONID`) does not have the security attribute set `Secure` .

Solution: check and adjust `secure` attribute in the file server.xml (https://tomcat.apache.org/tomcat-9.0-doc/config/http.html) Tomcat application server, see above.

- `Missing HTTP Strict Transport Security policy`

The application does not set the HTTP header `Strict Transport Security` .

Solution: value Strict Transport Security headers (https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security) can be set in the configuration variable `strictTransportSecurity=max-age=31536000 ; includeSubDomains` , see. above.

- `Stored XSS cez SVG obrázok`

SVG file allows to embed JavaScript code in the body, in case of direct display of such file in the browser the JavaScript code is executed (in case of standard embedding via `img` the code is not executed and the display is safe).

Solution: restrict the ability to upload SVG files, see rights settings above. As protection WebJET CMS generates HTTP header for SVG files `Content-Security-Policy` with value `default-src 'self'` which prevents javascript code execution (https://github.com/digininja/svg_xss) when the image is displayed directly. The value is configurable via the configuration variable `contentSecurityPolicySvg` .

To verify the behavior, create an SVG file with the following content, upload it to the WebJET CMS and insert it into the test page and verify its views and (non-)execution of the XSS attack:

```xml
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg"> <polygon id="triangle" points="0,0 0,100 100,0" fill="#3e70b7"
stroke="#004400"/>
<script type="text/javascript">
alert("SVG XSS");
</script>
</svg>
```

- `Stored XSS` via modification of translation keys

Through the translation texts the editor is able to perform an attack of the type `Cross Site Scripting` against other visitors to the site or other administrators.

Translation keys are also used to insert HTML code (e.g. link to terms and conditions in the calculator, bolding, setting CSS styles), so this application technically allows you to insert HTML/JavaScript code as well (this is a feature, not a bug). Note that the editor can also insert JavaScript code directly in the page editor, there is no fundamental reason to prevent him from editing translation keys as well.

Solution: minimize the number of users with access to the Text Translation application. At the same time, users who do not have the "Edit texts - view all texts" permission have limited editing options. They can only edit selected keys (set via conf. variable `propertiesEnabledKeys`) and at the same time the modified value is filtered and allows only defined HTML tags and attributes. These are set in the conf. variable `propAllowedTags` where marks are permitted by default `p,div,a,sub,sup,br,strong` and attributes in the conf. variable `propAllowedAttrs` where attributes are allowed by default `href,src,style,class,rel` . If you want to completely prevent the possibility of inserting HTML code for users without the "Edit texts - display all texts" permission, you can set the conf. variable `propAllowedTags` to an empty value (or a character `-`). By setting it to the character `*` the protection is switched off.

We also recommend setting up notifications when you change to a security engineer as an additional protection, see above.

- `Insecure Deserialization`

The web page import contains .xml documents with serialized java objects. However, these .xml documents can be modified and served with their own serialized java objects of type `<object class="java.lang.Runtime" method="getRuntime">` that execute the specified

operation directly on the server.

For a successful exploit it is necessary to modify the conf. variable `XMLDecoderAllowedClasses` that contains the list of allowed deserialized objects and add there the value `java.lang.Runtime` .

Solution: a normal user must not have permissions to modify configuration variables, only a dedicated person should modify them. As an additional protection, we have added directly into the code (without the possibility to edit this list) the unauthorized object types that cannot be added (enabled) through the configuration.

- `Cookies: Set the 'SameSite' flag as a counter measure to cross-site request forgery`

For `cookies` attribute can be set SameSite (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie/SameSite) to prevent cookies from being sent in a CSRF attack (to prevent cookies from being sent to other domains).

Currently in `servlet-api` support for setting this value in version 5.x is forthcoming, which may mean a long wait for direct programming support. However, when using Apache Tomcat, it is possible to set this value in the configuration using CookieProcessor (https://tomcat.apache.org/tomcat-8.5-doc/config/cookie-processor.html), which in the standard implementation allows the value to be set:

```xml
<Context>
    ...
    <CookieProcessor sameSiteCookies="strict"/>
</Context>
```

2. Vulnerability scanning of libraries

Using the tool OWASP Dependency-Check (https://jeremylong.github.io/DependencyCheck/index.html) you can easily check for vulnerabilities in the Java and JavaScript libraries of a web application. We recommend checking these on a regular basis.

If you have access to the source code/gradle of the project you can run the analysis directly using gradlew command (https://docs.webjetcms.sk/latest/en/developer/backend/security).

But the tool can also be run over the generated `war` web application archive. Install the version of the tool for command line (https://jeremylong.github.io/DependencyCheck/dependency-check-cli/index.html).

You can then run the check using the command:

```sh
dependency-check --project "Meno projektu" --suppression dependency-check-suppressions.xml --suppression
dependency-check-suppressions-project.xml --scan build/libs/*.war
```

parameters are set:

- `--project` - the name of the project that will be displayed in the report.
- `--suppression` - way to file with exceptions (https://docs.webjetcms.sk/latest/en/developer/backend/security), typically this file is part of a git repository.
- `--scan` - the path to the file/directory to be analyzed.

The result is a set of `dependency-check-report.html` in the current directory.

3. WebJET update

The WebJET Update section is used to update WebJET versions. The left part of the screen shows a list of available WebJET versions to which your WebJET can be updated. The current version of your WebJET is marked with an icon in the list ✅.

When you click on each version, a list of all the changes that the selected version brings will be displayed on the right side of the screen.
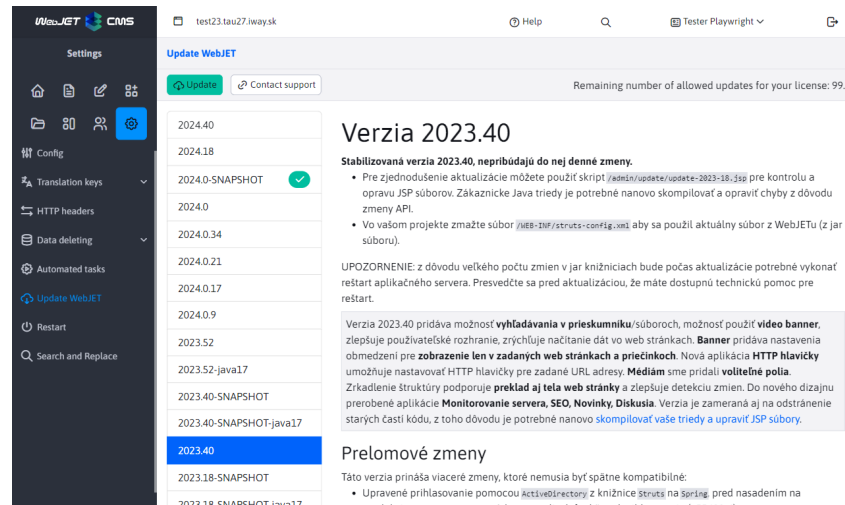
⚠ **Warning:** only update WebJET if you know what you are doing. Contact your hosting provider for support before updating. It may happen that WebJET does not boot properly after the update and a server restart will be required.

If your project contains additional JAR libraries you need to place them in the folder `/WEB-INF/lib-custom/`. The folder is fully replaced during the update `/WEB-INF/lib/` and thus your libraries would be deleted. This may result in an inability to boot after a reboot. If such a situation occurs copy the missing libraries to `/WEB-INF/lib/` from the advance.

3.1. Update to a specific version

To upgrade WebJET to a specific version, you need to select the desired version and then use the button to start the upgrade

3.2. Updating from a file

The ability to use update from file is also supported. This option can also be selected in the left menu as

.
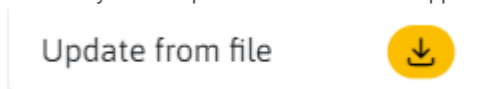
You will then be prompted to select and then upload the file using the **Upload file**. If the file is successfully uploaded, you will be prompted to start the update itself using the .

4. Audit

The Audit application is a tool for tracking changes in the system. The Audit -> Search section allows you to view and filter individual recorded values. Filtering is possible by type of audit records, time, user, etc.

5. Types of audit records

Each audit record automatically records the date and time, the ID of the logged-on user, the IP address, and, if the reverse DNS server is enabled, the computer name. The cluster node name, URI address, domain, and the value of the User-Agent http header are automatically inserted into the text of the audit record.

- `ADMINLOG_NOTIFY` - change in the list of notifications in the Audit application.
- `BANNER` - operations in the Banner System application
- `BASKET` - operations in the E-Commerce application
- `CALENDAR` - operations in the Event Calendar application
- `CONF_DELETE` - deleting a configuration variable, records its name

- `CONF_UPDATE` - changing or adding a configuration variable (in the Settings section), records the name, current value and new value of the variable
- `COOKIE_ACCEPTED` - acceptance of cookies on the website
- `COOKIE_REJECTED` - refusal to use cookies on the website
- `CRON` - logs background jobs running if the Audit option is checked. It also saves errors during task execution (if any), in which case it logs `Stack Trace` .
- `DATA_DELETING` - records the execution of the data deletion in Settings-Data deletion. Records the key that was deleted in the cache, or `ALL` to delete everything. When deleting the image cache, it records the path to the directory. When deleting the persistent cache, records the record ID.
- `DMAIL` - Bulk email application
- `DMAIL_AUTOSENDER` - used in the special situation of automatic sending of bulk email
- `DMAIL_BLACKLIST` - change in Bulk email->Unsubscribed emails
- `DMAIL_DOMAINLIMITS` - change in Bulk email->Domain limits
- `EXPORT_WEBJET` - not used
- `EXPORT` - data export operations (add, change, delete data export)
- `FILE_CREATE` - create a file or directory, record the path
- `FILE_DELETE` - deleting a file or directory, records the path
- `FILE_EDIT` - renaming, or editing a file, records the path
- `FILE_SAVE` - saving a file, e.g. when copying/moving it, etc. Record the path to the file
- `FILE_UPLOAD` - Uploading a file to WebJET, either via classic upload or Drag & Drop. It typically records the path to the uploaded file.
- `FORMMAIL` - submitting the form. It records the successful submission with a report `FormMail formName:` the name of the form, the list of beneficiaries and `referer` . On failure, it records the reason for not sending with a report `ERROR: formName:` name of the form, `fail:` reason for not sending. It also records spam detection by reporting `detectSpam TRUE:` reason for detection as spam.
- `FORM_ARCHIVE` - archiving the form, records the name of the form
- `FORM_DELETE` - form deletion, records the name of the form and possibly the ID if it is a deletion of a single record
- `FORM_EXPORT` - export of the form via the Export tab, currently universal export via buttons under the table is not recorded. The date of the last export is determined by this record for the ability to export since the last export.
- `FORM_REGEXP` - change in Forms->Regular expressions
- `FORM_VIEW` - not used
- `FORUM_SAVE` - detects the detection of vulgarity in the discussion forum
- `FORUM` - operations in the Discussion app
- `GALLERY` - changes in the Gallery app - creating a directory, adding/deleting a photo
- `GDPR_FORMS_DELETE` - GDPR application, deletion of old forms
- `GDPR_USERS_DELETE` - GDPR application, deleting old users
- `GDPR_BASKET_INVOICES_DELETE` - GDPR application, deleting old orders from e-commerce
- `GDPR_EMAILS_DELETE` - GDPR application, deleting old emails
- `GDPR_REGEXP` - GDPR application, regular expression management
- `GDPR_DELETE` - GDPR application, data deletion settings
- `GDPR_COOKIES` - GDPR application, cookie management
- `GROUP` - create/save/delete a directory in the Web pages section
- `HELPDESK` - not used
- `HELP_LAST_SEEN` - is used to record the date the What's New section is displayed in the help. When logging in, this section looks for the most recent file and compares it against the recorded date in Audit. If there is a newer file, a pop-up help window with the What's New section will appear after logging in.
- `IMPORTXLS` - Excel file import, used in customer implementations. Records the path to the imported file and its size
- `IMPORT_WEBJET` - not used
- `INIT` - WebJET initialization (start), records the path to the directory in which WebJET was started on the application server and the WebJET version number
- `INQUIRY` - operations in the Poll app
- `INQUIRY` - adding a question in the Poll app, records the text of the question
- `INSERT_SCRIPT` - change in the Scripts application
- `INVENTORY` - operations in the Property application
- `JSPERROR` - error when executing a JSP file when displaying a web page, logged `Stack Trace` errors

- `MEDIA` - Media operation (Media tab in the web page).
- `MEDIA_GROUP` - Media group management application.
- `PAGE_DELETE` - delete, move to trash, or request to delete a page, records the page ID
- `PAGE_UPDATE` - records changes in the page outside the standard saving in the editor - bulk operations in the page list
- `PAYMENT_GATEWAY` - calling the payment gateway in the E-Commerce application
- `PEREX_GROUP_CREATE` - create a perex group, record its name
- `PEREX_GROUP_DELETE` - deleting a group's perex, records its name and ID
- `PEREX_GROUP_UPDATE` - change the perex of a group, record its name
- `PERSISTENT_CACHE` - change in Data deletion->Persistent cache objects
- `PROP_DELETE` - deleting the translation text, records the language and key
- `PROP_UPDATE` - editing the translation text (in the Settings section), records the language, key and value
- `PROXY` - proxy application operations
- `QA` - operations in the Questions & Answers app
- `REDIRECT_CREATE` - create a new redirect (url or domain)
- `REDIRECT_DELETE` - deleting a redirect (url or domain), records the source and, for the domain, the destination of the redirect
- `REDIRECT_UPDATE` - change redirection (url or domain), record source and destination address
- `RUNTIME_ERROR` - logs a missing template for page view
- `SAVEDOC` - saving the web page in the Editor, it also records requests for approval. Records page title, page ID and ID in history
- `SENDMAIL` - sending an email (outside of forms), records the sender's email, the recipient's email and the subject of the email
- `SE_SITEMAP` - file generation `/sitemap.xml` , records the ID of the directory for which the sitemap is being generated and the contents of the User-Agent header
- `SQLERROR` - database error, logs the SQL error, the source of the error and `Stack Trace`
- `TEMPLATE_DELETE` - deleting a template, records the name of the deleted template
- `TEMPLATE_INSERT` - create a new template, record its name
- `TEMPLATE_UPDATE` - change in template, records list of changed fields
- `TEMPLATE_GROUP` - change in template group
- `TIP` - operations in the Tip of the Day app
- `TOOLTIP` - change in the Tooltip app
- `UPDATEDB` - making changes to the database via the admin console
- `USER_AUTHORIZE` - user authorization (approval of access to the password-protected section). Records the ID of the deleted user, if the user's ID is also known `login` and name.
- `USER_CHANGE_PASSWORD` - audits the user's password change. Based on the date, the password change interval is calculated (if set)
- `USER_DELETE` - deleting a user. Records the ID of the deleted user, if known, and the user's `login` and name.
- `USER_EDIT` - records the user's edit open event, it is not yet a save. Logs the user ID, `login` and name.
- `USER_GROUP_DELETE` - delete a user group, record the group name and its ID
- `USER_GROUP_INSERT` - create a new user group, record the name of the new group and its type
- `USER_GROUP_UPDATE` - save a group of users, record the name of the group and a list of changes
- `USER_INSERT` - creating a new user (or a new registration in a password-protected section). Record the user ID, `login` and name.
- `USER_LOGOFF` - logging out a user by clicking on the logout icon, it records the login name and information about whether the user is an administrator or a registered visitor
- `USER_LOGON` - user login, records the login name and information about whether the user is an administrator or a registered visitor. It also logs an invalid password event if the user is not authorized or the login name is unknown
- `USER_PERM_GROUP` - operations with rights groups, records the name of the group and, when changed, the list of changes
- `USER_SAVE` - records changes to the user in a password-protected section (if it contains a form for changing data)
- `USER_UPDATE` - saving an existing user. Records the current rights settings and changes to the entered data
- `WEB_SERVICES` - customer calls `WebServices` (usage depends on the implementation for a specific customer)
- `XSRF` - XSRF attack on the server (unauthorized referer header), logs the domain name value from `referer` Headers
- `XSS` - XSS attack on the server or a direct (unauthorized) call to a JSP file. Logs the URL or expression for which the attack was evaluated (e.g., unauthorized token in the URL, unauthorized HTTP method). Also logs cookie stealing (session IP address change).

6. Special audit format

If necessary, code can be added to WebJET that will save audit records to a special file or send them to a designated service. It is necessary to set the conf. variable `adminlogCustomLogger` to a Java class that implements the class `sk.iway.iwcm.AdminlogCustomLogger`. For each entry, the method is called `addLog(logType, requestBean, descriptionParam, timestamp)`

7. List of notifications

In the menu item Notification list you can set up email notifications for certain system events/errors. We recommend to set notification for events of the type `XSS` a `SQLERROR`.
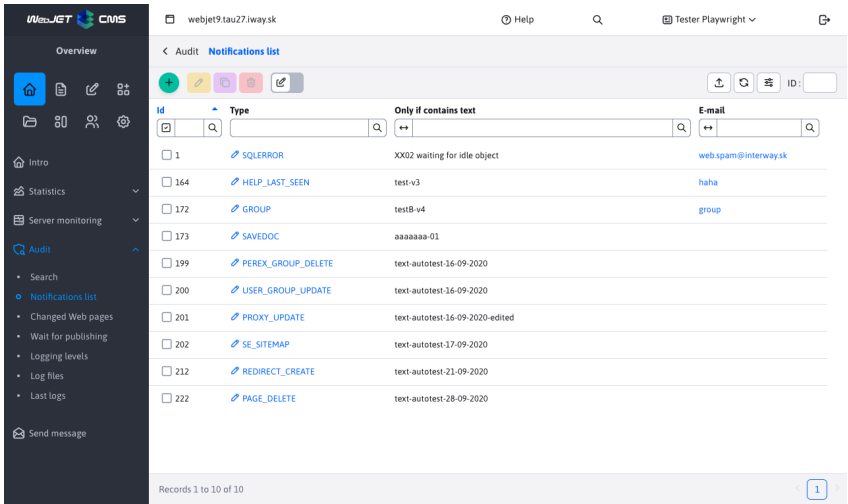
In the editor, you can also set the additional text that the error must contain to be sent to the specified email.
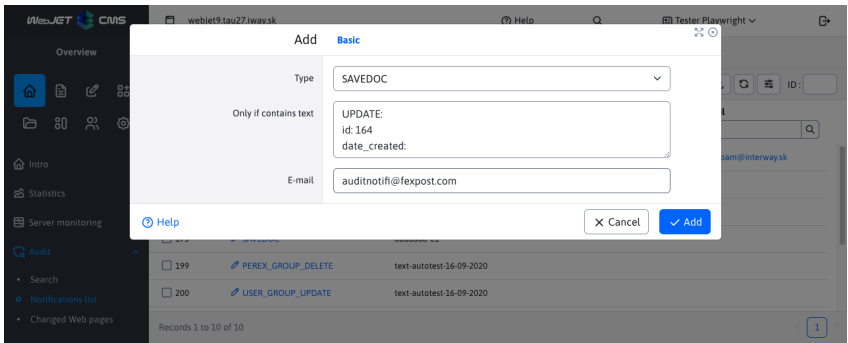
8. Changed pages

In the menu item Changed Pages you can see a list of changed stalls, sorted from last changed. You can work with these pages in the same way as in the List of web pages (https://docs.webjetcms.sk/latest/en/redactor/webpages/README).

All pages are displayed regardless of the user's rights to the page tree structure and the selected domain.

9. Awaiting publication

The Pending menu item lists the pages that are scheduled to be published in the future. It contains information about what web page (at what address/path) and when it will be published. For more information about the web pages, please click here List of web pages (https://docs.webjetcms.sk/latest/en/redactor/webpages/README).

The goal is to clearly display a list of pages that will be automatically changed in the future. All pages are displayed, regardless of the user's rights to the page tree structure and the selected domain.

Web pages that are waiting for display to be turned off are shown in red - they are set to Publish page after this date. Such pages will not be publicly viewable after the set date.

The list does not show the pages in the trash, no publishing is done for those.

10. Logging levels

The Logging Levels application allows you to manage the logging levels for individual java packages.
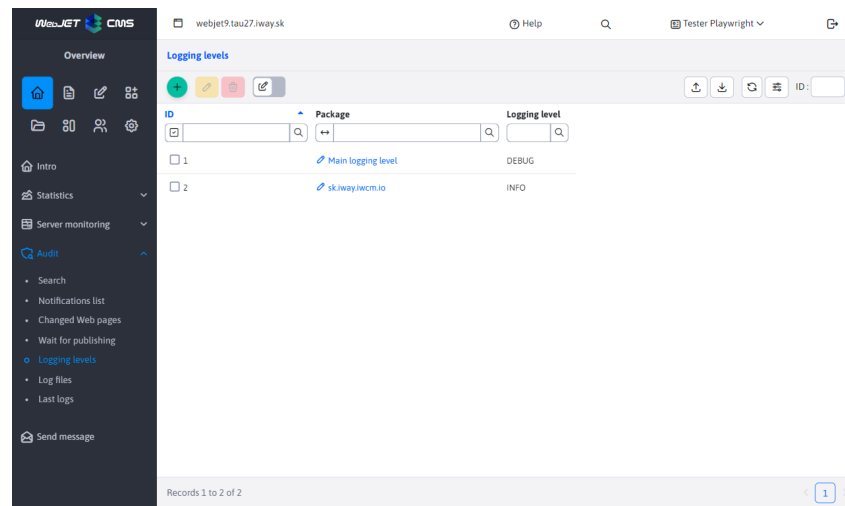
The first record in the table is always **Main logging level** (basic level).

2 configuration variables are used:

- **logLevel**, contains the logging level value for **Main logging level**
- **logLevels**, contains a list of java packages with logging levels (each on a new line). For example:

```
sk.iway=DEBUG
sk.iway.iwcm=WARN
org.springframework=WARN
```

Changes above the table are stored locally in a constant. If you want to save the changes (settings) permanently, in the editor you need to select the option **Save to database**. When saved, the configuration variables in the database are updated.

11. Add

For add actions, the java package value and logging level are required. If you specify an already added package, a duplicate value is not created but the existing one is updated.

12. Edit by

The editing action behaves differently for the Main logging level and other logging levels.

12.1. Main logging level

When editing the main level, we can only select NORMAL or DEBUG logging (for detailed logging). If you change the value in the editor `Package` , no change shall be made. Since the main level must still be present, only the logging level value can be changed.

12.2. Other logging

The logging level change will be saved, if you change the package, the original logging will disappear and will be replaced by this new one. All logging levels are allowed except the NORMAL value.

13. Lubrication

All logging level packages can be deleted except **Main logging level**. When you try to delete it, nothing will happen to it (not even the value will change).

14. Log files

The application provides an overview of all log files. Modifications above the table are not allowed. The table is for overview only. In the top left of the page, you can see the path where these files are stored.

When you click on the file name, a window with the contents of the file is displayed. The header contains fields for the ability to filter through the file using regular expressions (or directly by search term only).
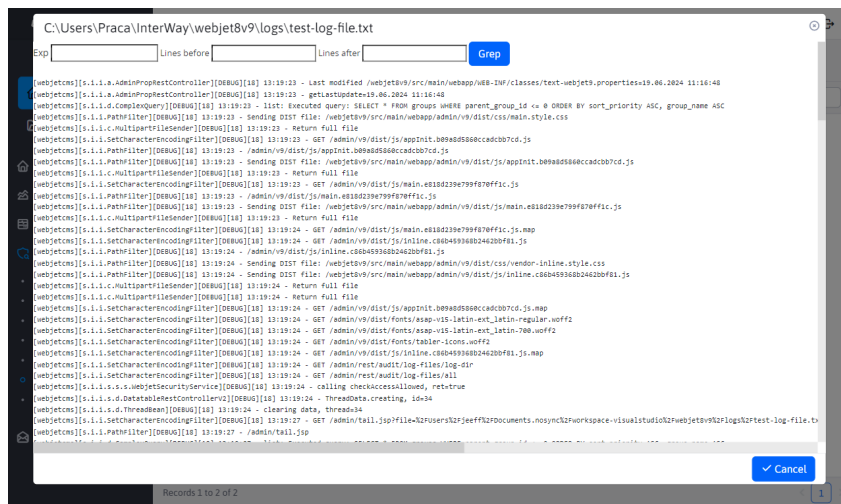
15. Latest logs

The application is designed to display the most recent logs, in case you do not have access to the logs on the file system. It displays logs that pass through the log framework (i.e. use the class `Logger`), does not display logs written directly via `System.out` or `System.err` .
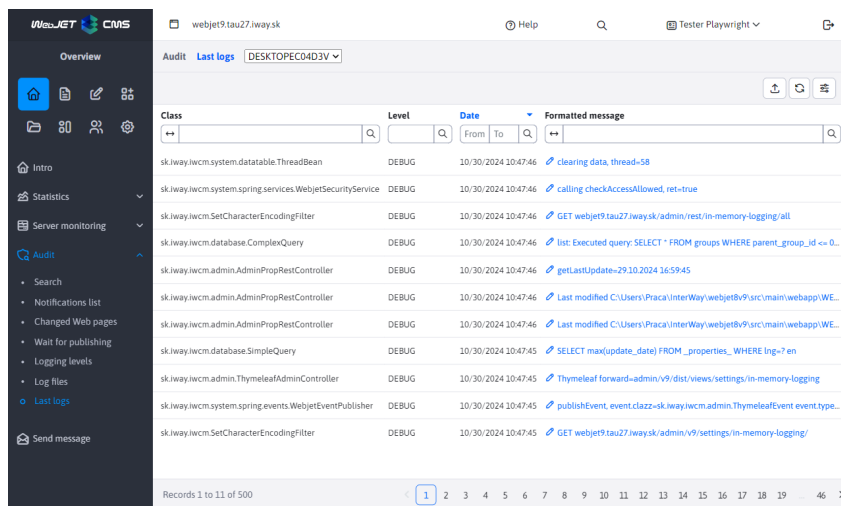
It supports clustering, so it is possible to request the last logs from another node of the cluster. In the tab `Stack Trace` stack dump is located (but the contents are only displayed for error logs, for standard log levels it is empty).

15.1. Configuration options/settings:

- `loggingInMemoryEnabled` - by setting it to `true/false` Enable or disable log storage.
- `loggingInMemoryQueueSize` - maximum number of logs written to memory (default 200). Please note that all data is loaded into the table at once and due to the transfer `stack trace` can be large. We do not recommend setting this variable to an extremely high value.

To function properly, it must be `logger` also set in the file `logback.xml` . By default it is set this way, but if you have changed the file, you need to add `IN_MEMORY appender` and add his call for `root` element.

```xml
...
<appender name="IN_MEMORY" class="sk.iway.iwcm.system.logging.InMemoryLoggerAppender" />


<root level="ERROR">
    <appender-ref ref="STDOUT" />
    <appender-ref ref="IN_MEMORY" />
```

```
    </root>
    ...
```

15.2. Implementation details

- `sk.iway.iwcm.system.logging.InMemoryLoggerAppender` - `appender` For `logback` , which ensures that logs are sent to `InMemoryLoggingDB`
- `sk.iway.iwcm.system.logging.InMemoryLoggingDB` - class provides writing and retrieving logs from and to `queue` , retrieving logs on the cluster
- `sk.iway.iwcm.system.logging.InMemoryLoggingEvent` - model for log event
- `sk.iway.iwcm.system.logging.InMemoryLoggerRestController` - controller for log dump to DataTable

16. Server performance

For optimal server performance, several requirements and settings must be met. Each application (e.g. photo gallery, poll, etc.) embedded in a web page causes a slowdown. Applications typically make additional database requests or need to read data from the file system.

Search engines that constantly crawl and index web pages on your server can also have a significant impact on performance. Their traffic may not be visible e.g. in Google Analytics, but it is visible in Statistics (https://docs.webjetcms.sk/latest/en/redactor/apps/stat/README) provided by WebJET CMS.

16.1. Identification of problems

The first thing to do is to identify where the slowdown is occurring. If you can identify at a glance a web page that seems slow to you, you can use the URL parameter `?_writePerfStat=true` . Otherwise, turn on server monitoring in which you can identify the web pages that are taking the longest to execute.

16.1.1. URL parameter

Using the URL parameter `?_writePerfStat=true` it is possible to get a list of applications embedded in the web page with the time of their execution. For example, a page `/sk/` view as `/sk/?_writePerfStat=true` .

When displaying a web page in this way, an expression of the type `PerfStat: 3 ms (+3) !INCLUDE(...)` . It may not be easily searchable in a standard web page, so we recommend to view the source code of the page - in Chrome menu View-Developer-View Source Code. Then use the browser search term `PerfStat:` .

This expression is in the format `PerfStat: 3 ms (+3)` where the first number is the total execution time of one `iwcm:write` expression and the number in parentheses is the execution time of this application. This is followed by the path to the application and its parameters. So you are interested in the primary number in parentheses.

Using the URL parameter `_disableCache=true` you can turn off application caching.

16.1.2. Server monitoring

For a comprehensive view, you can turn on the server monitoring by setting the following configuration variables:

- `serverMonitoringEnable` - enables the server monitoring and logging function
- `serverMonitoringEnablePerformance` - turns on application and website performance monitoring
- `serverMonitoringEnableJPA` - enables the SQL query monitoring function

> ⓘ **Warning:** application performance monitoring and SQL query monitoring puts a strain on the server, we do not recommend to have this feature permanently enabled.

After setting the configuration variables, you need to perform **restart the application server** to activate performance monitoring on initialization.

Then, in the Server Monitoring - Applications/WEB Pages/SQL Queries section, you can identify the parts that are taking a long time to execute. Focus on the most frequently executed applications/SQL queries and optimize them.

16.1.3. Total web page generation time

There is an app `/components/_common/generation_time.jsp` which, if inserted into the footer of the web page template, will generate the total time of web page generation into the HTML code.

The following application parameters can be set:

- `hide` - default `true` - the generation time is displayed as a comment in the HTML code
- `onlyForAdmin` - default `false` - generation time is displayed only if an administrator is logged in

Insert the following code into the footer (or a suitable free field) of the web page template:

```html
!INCLUDE(/components/_common/generation_time.jsp, hide=true, onlyForAdmin=false)!
```

At the location of the embedded application, information about the execution time of the entire web page in ms is displayed:

```html
<!-- generation time: 4511 ms -->
```

16.2. Measuring database server and file system performance

To compare the performance of environments - e.g. test VS production environments, the scripts below can be used. Running them requires the right to update WebJET. You can measure and compare environments without load, but also during operation or performance tests.

- `/admin/update/dbspeedtest.jsp` - measures the performance of reading data from the database server.

Good values are for example:

```html
Image read, count=445
...
Total time: 649 ms, per item: 1.4584269662921348 ms
Total bytes: 4.8050469E7, per second: 7.403770261941448E7 B/s


Random web page read, count=3716
...
Total time: 3608 ms, per item: 0.9709364908503767 ms
Total bytes: 1371566.0, per second: 380145.78713968955 B/s
```

```
Only documents.data web page read, count=3716
...
Total time: 2205 ms, per item: 0.5933799784714747 ms
Total bytes: 685783.0, per second: 311012.6984126984 B/s


Documents read using web page API, count=3716
...
Total time: 1869 ms, per item: 0.5029601722282023 ms
Total bytes: 685783.0, per second: 366925.09363295883 B/s
```

Due to the different number of records in the database, it is necessary to compare `per item` Values.

- `/admin/update/fsspeedtest.jsp` - checks the speed of reading a list of files from the file system, it should be checked especially if you are using a network file system.

Good values are for example:

```html
Testing mime speed, start=0 ms
has base file object, fullPath=/Users/jeeff/Documents.nosync/workspace-visualstudio/webjet/webjet8v9-
hotfix/src/main/webapp/components/_common/mime diff=1 ms
listFiles, size=678, diff=284 ms
listing done, diff=16 ms


Testing modinfo speed, start=0 ms
modinfo list, size=102, diff=1 ms
modinfo listing done, diff=220 ms
Total time=522ms
```

16.3. Database query optimization

To optimize the number of database requests, you can enable caching - `cache` .

16.3.1. Web pages

Each web page has an option in the Basic tab **Enable page caching**. Turning this option on takes the web page content from the table `documents` is cached. When a web page is displayed, it will not be necessary to make a database call to retrieve the contents of the web page.

We recommend enabling this option on the most visited websites, which you can get a list of in the app Statistics (https://docs.webjetcms.sk/latest/en/redactor/apps/stat/README).

16.3.2. Applications

Similar to web pages, you can also enable caching for applications. Some applications have this option available directly in application settings (https://docs.webjetcms.sk/latest/en/custom-apps/appstore/README) embedded in the web page in the View as field tab **Buffer time**.

If the application does not have this setting available you can still set the parameter in the HTML code of the web page text by adding the parameter `, cacheMinutes=xxx` to the parameters of the embedded application, for example:

```html
!INCLUDE(sk.iway.iwcm.components.reservation.TimeBookApp, reservationObjectIds=2560+2561, device=,
cacheMinutes=10)!
```

> ❗ **Warning:** it is important to note that the cache is global for the entire application server. The application file path, the individual parameters specified in the HTML code of the web page, and the language of the currently displayed web page are used as the key. The URL parameters of the web page are not taken into account.

Thus, the cache cannot be used if, for example, a paging list is displayed where the page number is passed using a URL parameter. However, there is an exception for applications containing a list of news items in the filename `/news/news` the buffer is used only if no parameter is specified in the URL `page`, or the value of this parameter is different than `1`. In this way, the buffer is also used for the news list, but only the first page of results is saved. Other pages are not saved.

16.4. File system optimization

Web pages typically contain many additional files - images, CSS stylesheets, JavaScript files, and so on - that need to be loaded along with the web page. Therefore, the display speed also depends on the number and size of these files.

16.4.1. Setting the buffer

It is possible to set the browser to use a cache for web page files - this way the file will not be read repeatedly each time the web page is viewed, but if the browser already has it cached, it will be used. This will speed up the web page display and reduce the load on the server. An example is the logo image, which is typically on every page, but is highly unlikely to change - or changes on the order of once every few months.

It is possible to set the following configuration variables that affect the HTTP header `Cache-Control` :

- `cacheStaticContentSeconds` - set number of seconds, default `300` .
- `cacheStaticContentSuffixes` - a list of extensions for which the HTTP header `Cache-Control` generated, by default `.gif,.jpg,.png,.swf,.css,.js,.woff,.svg,.woff2` .

For a more precise setting, you can use the app HTTP headers (https://docs.webjetcms.sk/latest/en/admin/settings/response-header/README) where you can set different values for different URLs.
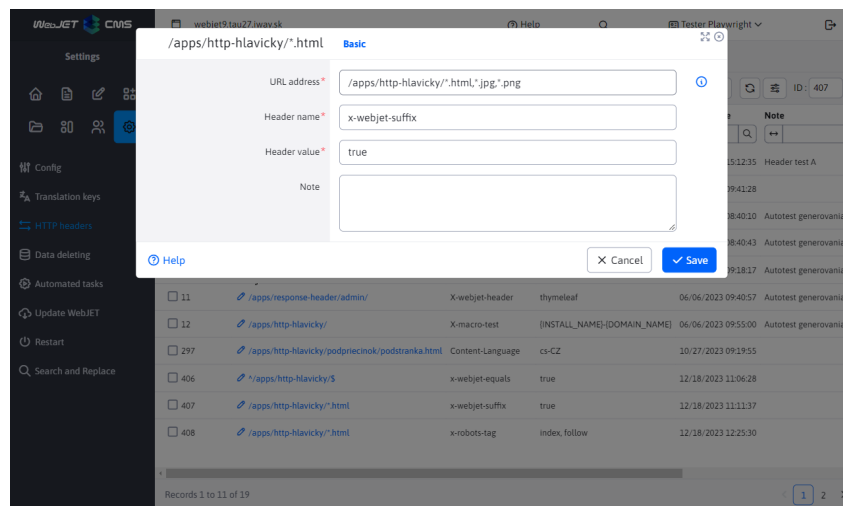
16.5. Behaviour for the administrator

If an administrator is logged in the application buffer is not used (it is assumed that the administrator always wants to see the current state).

This behaviour can be changed by setting the configuration variable `cacheStaticContentForAdmin` to the value of `true` . It is particularly appropriate to set this value for intranet installations where users authenticate against `SSO/ActiveDirectory` server and even when working in the intranet environment they have administrator rights.

16.6. Search engines

Search engines and various other bots can put a significant load on the server. Especially with the advent of AI learning, there is significant internet crawling and database populating for AI learning. Bots often try different URL parameters to retrieve additional data.

16.6.1. Setting robots.txt

The behaviour of the robots can be influenced by settings in the file `/robots.txt` . This if it does not exist is generated by default. Place your modified version in `/files/robots.txt` , from this location WebJET will display it when calling `/robots.txt` .

Using the file [robots.txt (https://en.wikipedia.org/wiki/Robots.txt)](https://en.wikipedia.org/wiki/Robots.txt) you can influence the behaviour of robots and search engines - limit the URLs they can use, set the spacing between requests, etc.

16.7. Other settings

16.7.1. Reverse DNS server

Statistics, auditing, and other applications can retrieve the reverse DNS record from the IP address. API calls are used `InetAddress.getByName(ip).getHostName()` . However, the DNS server may not be available on the servers/DMZ and this call may take a few seconds before an error occurs. Generally such a call slows down the execution of the HTTP request.

By setting the configuration variable `disableReverseDns` to the value of `true` it is possible to disable DNS name retrieval from the visitor's IP address and speed up the execution of queries. In the field for the value `hostname` the value of the IP address is then written.

16.7.2. Turning off statistics

Writing statistics data is asynchronous, it is done in batches so that the web page display does not wait for the statistics data to be written to the database.

When traffic is high, or you are looking for performance issues, you can temporarily disable the logging of traffic statistics by setting the configuration variable `statMode` to the value of `none` . The standard value is `new` .

17. Server monitoring

17.1. Internal monitoring

The performance and load analysis of the server, individual applications, database queries and the pages themselves can be monitored directly in the Server Monitoring application (in the WebJET administration in the Overview section).

The module provides the following options:

- **Current values** - the current server load, memory, and number of database connections.
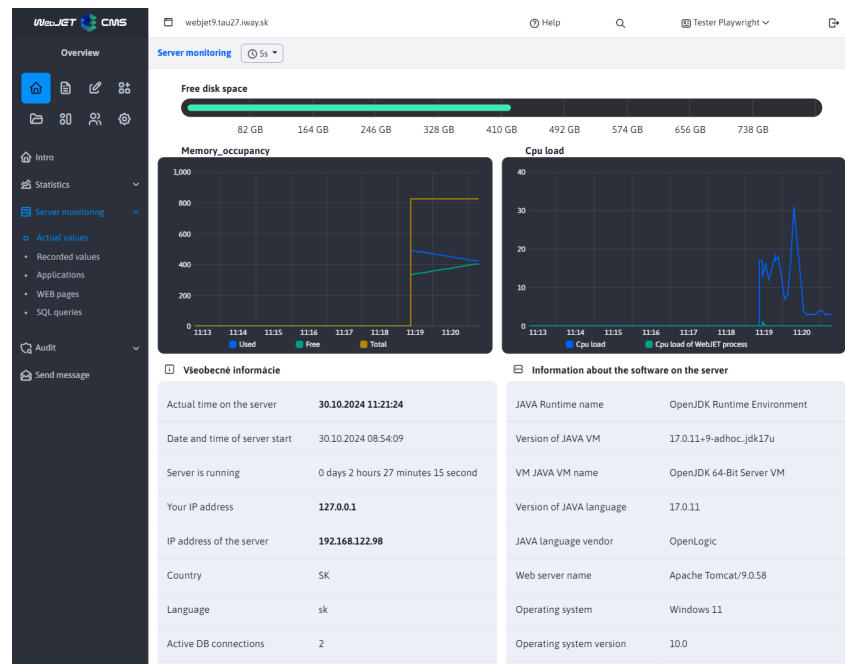
- **Recorded values** - a listing of historical recorded memory usage values, `sessions`, cache and database connection. For storing historical values it is necessary to set the conf. variable `serverMonitoringEnable` to the value of `true`.
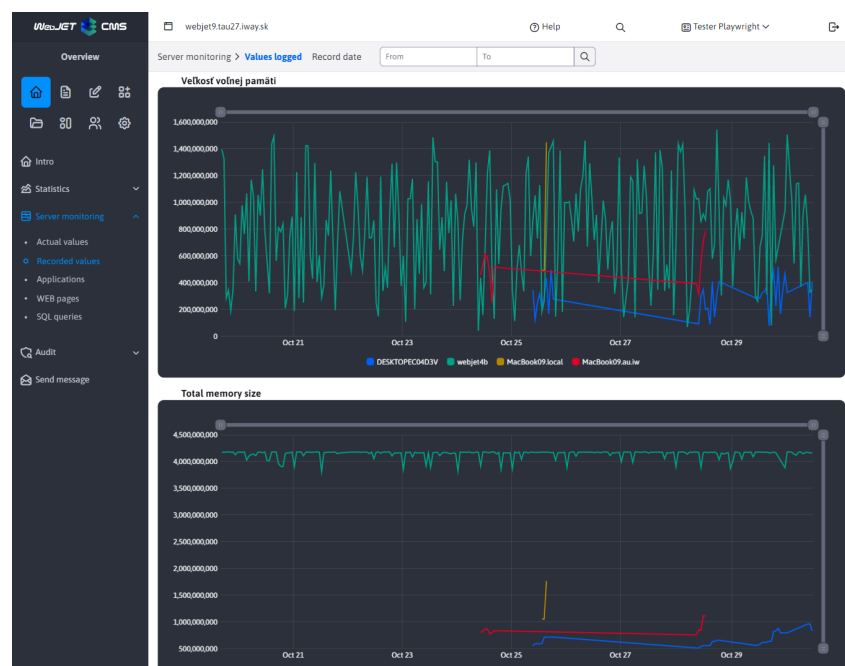
After setting the configuration variable `serverMonitoringEnablePerformance` at `true` are also available:

- **Applications** - statistics on the execution of individual applications. Shows the number of executions, average execution time, number of executions from cache memory, and slowest execution.
- **WEB pages** - statistics of individual web page views. It shows the number of views, average view time, slowest and fastest views.

After setting the configuration variable `serverMonitoringEnableJPA` at `true` is also available:

- **SQL queries** - SQL query execution speed statistics. It shows the number of executions, average execution time, slowest and fastest execution and the SQL query itself.

> ⚠ **Warning:** Activating monitoring affects server performance and memory load. In addition to the ability to log values, enabling monitoring has an impact on server performance. All data except for the logged values section is held only in the server's memory, so it will start logging again when the server is restarted.

> ⚠ **Warning:** modular options **Applications**, **WEB pages** a **SQL queries** use a unique common logic, which is described in more detail in Server monitoring by selected node

17.2. Remote monitoring of the server runtime

If you need to monitor the status of WebJET via Nagios (http://www.nagios.org) /Zabbix (https://www.zabbix.com) or other service, WebJET provides at the URL `/components/server_monitoring/monitor.jsp` your condition. HTTP Responses **state 200 if everything is fine**, or **with a status of 500** (Internal Server Error) if **not all controls are met**.

This URL can also be called at one-second intervals, and we recommend using it within the cluster to monitor the availability of individual nodes.

Allowed IP addresses for which monitor.jsp responds correctly are set in the configuration variable `serverMonitoringEnableIPs` .

The component monitors the following parts:

- **WebJET initialization**, including its `preheating` (waiting for the initialization of cache objects or background tasks). The preheating time is set in the monitoringPreheatTime conf. variable (default 0). WebJET responds with the text `NOT INITIALISED` if it is not initialized correctly (e.g. there is no connectivity to the database at all when it starts, or it has an invalid license). Text `TOO SHORT AFTER START` responds during preheating time (inclusion in the cluster should wait for the background object/task cache to finish loading).
- Monitoring **availability of the database connection** - SQL select is performed from the table `documents` (specifically `SELECT title FROM documents WHERE doc_id=?`), while in the configuration variable `monitorTestDocId` is the docid of the tested page. If the SQL query fails it responds with the text `DEFAULT DOC NOT FOUND` .
- **Availability of templates** - if the list of initialized templates is less than 3 responds with the text `NOT ENOUGHT TEMPLATES` .
- **Recording of statistics data** - verifies that there are not suspiciously many records in the statistics write stack (the number of records is set in the configuration variable `statBufferSuspicionThreshold` , default 1000). If the statistics write stack contains a larger amount of data to write, this indicates either a SQL Server performance problem or a problem with background jobs. If the number of records is exceeded, it responds with the text `STAT BUFFER SUSPICION` .
- If it occurs **other error** replies with the text `EXCEPTION: xxxx` .

WebJET is also possible manually **switch to service mode** by setting the configuration variable `monitorMaintenanceMode` for real. Then monitor.jsp responds with the text `UNAVAILABLE` .

If all is well it replies with the text `OK` . For monitoring **it is sufficient to monitor the HTTP status** answers, the text is only informative for a more precise determination of the problem.

17.3. Configuration variables

- `serverMonitoringEnable` - if set to `true` , starts monitoring the server every 30 seconds and writes these values to the table `monitoring`
- `appendQueryStringWhenMonitoringDocuments` - capture SQL parameters during monitoring `?`
- `monitorTestDocId` - The ID of the page whose database connection (name retrieval) is being tested in the component `/components/server_monitoring/monitor.jsp` that can be tested by the surveillance SW (default value: 1)
- `serverMonitoringEnablePerformance` - if set to `true` , triggers speed monitoring of SQL queries, web pages and applications (default: false)
- `serverMonitoringEnableJPA` - if set to `true` , triggers SQL query execution speed monitoring for JPA, but results in an increase in server memory load (default: false)
- `serverMonitoringEnableIPs` - List of IP addresses from which the component is available `monitor.jsp` for server monitoring (default: 127.0.0.1,192.168.,10.,62.65.161.,85.248.107.,195.168.35.)

- `monitoringPreheatTime` - The number of seconds required for the web site to warm up (cache load) after a reboot, during which it will `monitor.jsp` component returning unavailability of cluster node (default: 0)
- `monitoringEnableCountUsersOnAllNodes` - If the public nodes of the cluster do not have the ability to write to the table `_conf_/webjet_conf` set to `false`. Total number of `sessions` will then only be available by summing the individual records in the server monitoring.

18. Restart

Click on the option **Restart** in the Settings section, you will see a confirmation to restart WebJET. The reboot will be performed on the server by default, but it depends on the server settings whether the reboot from the web application is enabled. If not, the restart will not be performed.

In the configuration `server.xml` application server Tomcat (https://tomcat.apache.org/tomcat-9.0-doc/config/context.html) it is necessary to have restart enabled using the attribute `reloadable="true"` in the elements `Context` :

```xml
<Host name="...">
    <Context reloadable="true" />
</Host>
```

> ⛔ **Warning:** Before restarting, check the availability of your hosting's technical support, as the application server may need to be restarted as well. However, this cannot be done directly from the WebJET environment.

Repeated restarts can also fill the application server's memory, which may require restarting the application server directly on the server.

19. Exchange cluster node data

Website **Applications**, **WEB pages** a **SQL queries** share the same logic regarding server monitoring according to the currently selected node. To select a node, use the field that appears in the page header next to the page name.

When opened by clicking, we can see all the available options. The default value is always the current node (the node of the cluster you are currently logged in to), which is marked with the text `(Aktuálny uzol)` .
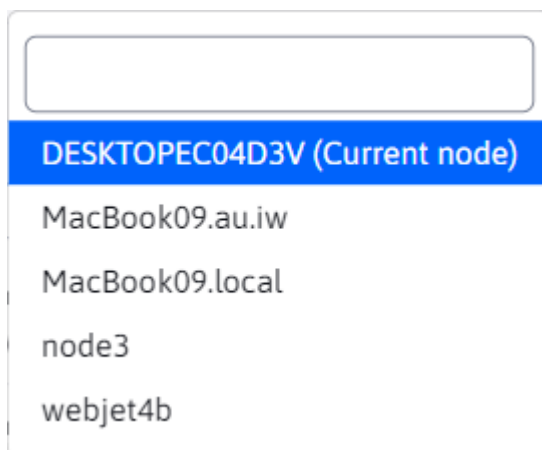
If the selected node is current, the locally stored data is displayed. In this case, there is also a delete button that deletes this locally stored data (the delete button is only available for the current node). In the case of a node other than the current node, the data is retrieved from the database table.

19.1. Restore data - current node

If the current node is selected, pressing the data refresh button will only retrieve the currently stored data (database tables are not worked with here). If the data was previously deleted, it may take a while for new records to appear.

19.2. Data recovery - remote node

For nodes other than the current one, data recovery is more difficult. The data of other nodes is stored in a table `cluster_monitoring` . The data recovery process starts by deleting the data from the table, as it may no longer be up-to-date.
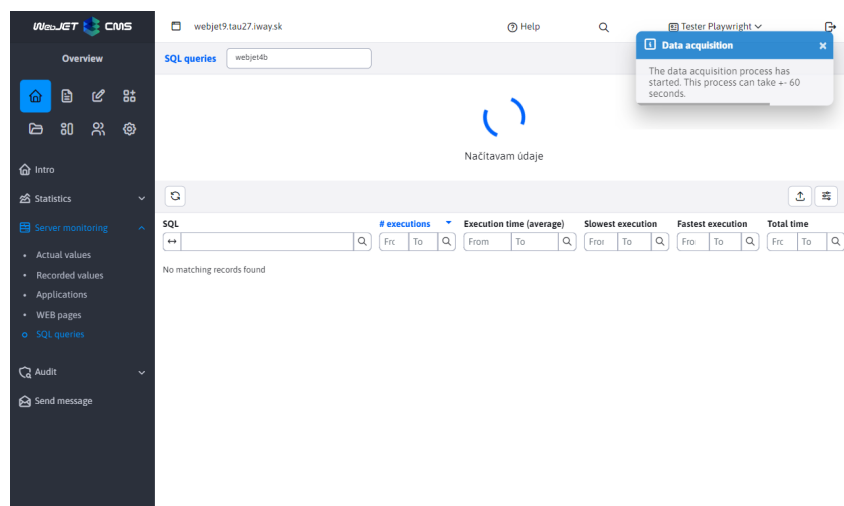
As you can see in the image above, the data has been removed and the animation waiting for the data is displayed. We also see an informational notification that warns us that this process may take +- a few seconds. This interval may vary depending on the configuration variable set `clusterRefreshTimeout` .

The process of retrieving current data consists of creating a request for current data for a node by creating a record in a database table `cluster_refresher` . The cluster itself in the intervals specified by the conf. variable `clusterRefreshTimeout` updates the data in the table `cluster_monitoring` for a specific node, if there is a request for that node in the table `cluster_refresher` . Therefore, the data retrieval process may take several minutes and may vary depending on the cluster refresh interval set (there may be a situation where the cluster interval was just before the refresh and the actual data is retrieved in 10 seconds, even though the interval was set to 5 minutes).

Although it is not displayed, the page will ask every 10 seconds if the table `cluster_monitoring` no new data has been added that could be displayed. If the requested node did not contain any data (but the table has already been updated), a new cluster request for the data will be created, and again we will check every 10 seconds to see if the data has already been updated. The whole process will repeat until the updated table `cluster_monitoring` will not contain at least one record to display. At that point, the animation is hidden and the currently retrieved data of the other node is displayed.

20. Data deletion

Application **Data deletion** allows you to remove unnecessary data from the database, which can increase server performance and free up disk space. You can find this tool in the **Settings** under the heading **Data deletion**.

20.1. Database records

Deleting data from selected database tables, deletion is possible from the following groups:

- **Statistics**: Removes statistical data. Deleting older data can significantly improve server performance, but you will lose information about site traffic for the selected period.
- **Emails**: Allows you to delete sent emails from the Bulk Email application and emails sent with a time delay (or emails sent within a multi-node cluster).
- **Site History**: Deletes the recorded historical versions of web pages, these are saved each time a web page is published. They are displayed in the History tab when editing a web page. Deleting does not affect the currently displayed pages, the historical versions are deleted.
- **Server monitoring**: Removes logged server monitoring data such as performance metrics and logs.
- **Audit**: Deletes audit records that monitor user activity and system events, only selected record types can be deleted.

With each deletion, an optimization of the database table is also performed to physically free up disk space and optimize the order of records in the database table.

20.2. Cache objects

Displays a list of objects stored in the application cache and allows you to delete them individually, which can reduce memory consumption or trigger a restore of data in the server cache. By clicking on the name, you can view the contents of the record for selected data types. The object is used for work Cache (https://github.com/webjetcms/webjetcms/tree/main/src/webjet8/java/sk/iway/iwcm/Cache.java)

20.3. Persistent cache objects

Managing and deleting objects stored in a persistent cache that retains data even after a server restart (data is stored in the database). The object is used for work PersistentCacheDB (https://github.com/webjetcms/webjetcms/tree/main/src/webjet8/java/sk/iway/iwcm/system/cache/PersistentCacheDB.java). Only text data can be stored in this cache, typically the method `downloadUrl(String url, int cacheInMinutes)` which downloads data from the specified URL in the background and updates it at the set time. The application uses this method and immediately retrieves the data from the cache.

21. System backup

The application is used to create a ZIP archive of individual folders of the WebJET file system. You can choose which folders to include in the ZIP archive and which folder to create the resulting ZIP archive in. A database backup is not created, this must be created with the database backup tools.

> **Warning:** The amount of data in the selected folders can be large and the ZIP file may not be generated correctly (the limitation is to a 2GB file). If necessary, you can create backups in parts (individual folders).

This process can take several tens of minutes depending on the amount of data in the selected folders. Wait for the whole process to finish. During this time, you should see information in the window about the number of pages already generated and the total number of pages.

The result is a zip archive created in the specified folder.