

Audits Management

This guide shows you how to manage Audits in Kiuwan Life Cycle.

- [What is a Kiuwan Audit?](#)
 - [Audit Checkpoints](#)
 - [Checkpoint types](#)
 - [Code Security and Code Analysis checkpoints](#)
 - [Insights components checkpoints](#)
 - [Other Audit and Checkpoints parameters](#)
 - [Checkpoint Weight](#)
 - [Audit Approval Threshold](#)
 - [Audit evaluation logic](#)
 - [Example 1](#)
 - [Example 2](#)
 - [Example 3](#)
- [Audit Management](#)
 - [Create an Audit](#)
 - [Checkpoint management](#)
 - [Create a Checkpoint](#)
 - [Associate an Audit to an Application](#)
 - [Setting a custom Audit as the default audit for new applications](#)
 - [How can I know which is the default audit?](#)
- [Audit Results](#)
 - [Audit Results Page](#)
 - [Audit Result when using Local Analyzer](#)
 - [Audit Result when using Kiuwan plugin for Jenkins](#)

What is a Kiuwan Audit?

The main purpose of an Audit is to evaluate if the results of an application delivery analysis satisfy a pre-defined set of conditions (checkpoints). Based on the results of that evaluation, the Audit will pass or fail.

In Kiuwan, you can pre-define as many Audits as you want as a set of checkpoints that are evaluated when the Audit is applied. These pre-defined Audits will be available in your account to be assigned to applications. The specific Audit assigned to an application is automatically applied to the results of any delivery analysis of that application.

For example, we have an application with a baseline analysis describing the actual state of the application (current defects and indicator level). And we want to define a corporate policy stating that any delivery (total or partial) of that application must not contain any new defect. In such a case, that delivery should not be accepted.

In this case, we can define an Audit to check that any delivery does not contain any new defect. In the case of a new defect, the Audit will FAIL, otherwise, it will be OK. This case is exactly what **Kiuwan's Default Audit** does, and the delivery will be marked as OK or FAIL depending on analysis results.

Similarly, we might define any other policies. Some examples might be:

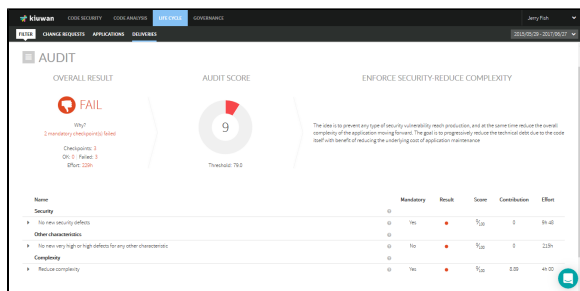
- **Enforce security:** there should not be any defect of Security rules (or very high priority security rules)
- **Enforce maintainability and efficiency:** there should not be any new defect related to high and very high maintainability and efficiency rules

All these Audits (and any others you might consider) can be defined in Kiuwan (without any programming) and will be applied automatically to every delivery analysis.

Kiuwan not only marks a delivery as OK or FAIL, but it will also specify:

- WHY (specific reasons for Audit failure),
- HOW to remediate it (i.e. the specific actions to be done to pass the Audit), and even
- HOW MUCH effort will take to pass the Audit

For every delivery analysis, Kiuwan provides a full Audit Report with all this useful information.



Audit Checkpoints

Kiwan Audits are based on **Checkpoints**.

A **checkpoint** is a specific (atomic) condition to be met by the analysis. An Audit may contain as many checkpoints as validations you want to check.

Every checkpoint has two possible **results**:

- **OK** (condition is met)
- **FAIL** (condition is not met)

Furthermore, depending on its level of compliance, a checkpoint can be classified as:

- **Mandatory**: something the delivery must meet to accept it or deploying in production
- **Optional**: something to check for, but not essential for accepting the delivery

An Audit will FAIL if any of its mandatory checkpoints fail. Please see [Checkpoint Management](#) for details on how to create and manage them.

Checkpoint types

Kiwan provides a **library** of checkpoint types you can use to define your specific checkpoints when creating an Audit.

Code Security and Code Analysis checkpoints

When defining your checkpoints, you will be able to define **thresholds** for:

- The maximum number of Total defects:
 - The total number of defects in the delivery must not be higher than the defined threshold.
- The maximum number of New defects:
 - The number of New defects (i.e. defects introduced by the delivery that don't exist in the baseline) must not be higher than the defined threshold.
- Some other metrics such as:
 - Global Indicator improvement, Global Indicator defined threshold, Duplicated Code threshold, Percentage of Very High defects, etc.

In addition, Kiwan not only allows you to define the number of defects and *Insights* components, but it also allows you to define the **nature or type of those defects**.

When selecting the nature or types of defects considered in a checkpoint, you can specify the following criteria:

- By **nature**: Languages, Characteristics and/or Priorities of found defects
 - For example, defects of high priority in Security and Maintainability in Java and Cobol
- By **type**: Defects of specific types
 - For example, defects found by a specific set of rules

Insights components checkpoints

When you define your checkpoints, you can also define **thresholds** for:

- The maximum number of total *Insights* components that have a defined security risk(s):
 - The total number of components in the delivery that meet the defined security risks **must not be higher** than the defined threshold.

You can also check if the discovered components meet specific criteria based on:

- Group: the group of the component to be found (e.g. com.fasterxml.jackson.core).

- Name: the name of the component to be found (e.g. jackson-databind).
- Version: the version number of the component to be found (e.g. 2.8.10).
- Comparator: the comparison operator to be used when finding components (allowed operators are: >, >=, =, <=, <).

Other Audit and Checkpoints parameters

Before explaining the logic applied during Audit evaluation, we need to define a couple of concepts and parameters you can control in the definition of Kiuwan Checkpoints and Audits.

Checkpoint Weight

Every checkpoint has an associated **Weight** that represents the relative weight of the checkpoint in the Audit.

The weights you specify translate (automatically) into a percentage contribution to the overall Audit.

For example, if your Audit has 2 checkpoints of equal importance, you should set this value to 1 for both, translating into a 50% contribution for each checkpoint.

Now, if you consider that one is 2 times more important than the other, you should set them as 2 and 1 respectively, translating into a 66% and a 33% contribution.

Audit Approval Threshold

In a Kiuwan Audit, you can specify an **Approval Threshold**.

This threshold will represent the minimum percentage of checkpoints contribution to consider the Audit as OK. Independently of whether they are mandatory or not, only the contribution percentage of each checkpoint is taken into account to evaluate this threshold.

Learn how the Audit evaluation logic works in the next section.

Audit evaluation logic



Remember: any checkpoint related to a baseline analysis, when such baseline does not exit, will automatically be OK as with a 100% of its contribution to the audit.

Examples of checkpoints related to a baseline analysis:

- New defects
- Global Indicator improvement

The logic behind **audit evaluation** is based on two-steps

1. **All mandatory checkpoints must be successful.** Otherwise, the Audit will FAIL.
2. If all the mandatory checkpoints are OK, the **sum of successful checkpoint percentage contribution** (based on the defined weights) **must be higher than the Audit Approval Threshold.** Otherwise, it will be FAIL.

Let's see this logic applied to some examples.

Example 1

Audit Approval Threshold = 75%

Checkpoint	Mandatory	Contribution	Result
A	Yes	50%	FAIL
B	No	30%	OK
C	No	20%	OK

The audit will FAIL. Mandatory checkpoint has failed, therefore the Audit result is FAIL.

Example 2

Audit Approval Threshold = 75%

Checkpoint	Mandatory?	Contribution	Result
------------	------------	--------------	--------

A	Yes	50%	OK
B	No	30%	FAIL
C	No	20%	OK

The audit will **FAIL**. Although the mandatory checkpoint is OK, the sum of successful checkpoints (70%) is lower than the Audit Approval Threshold (75%).

Example 3

Audit Approval Threshold = 75%

Checkpoint	Mandatory	Contribution	Result
A	Yes	50%	OK
B	No	30%	OK
C	No	20%	FAIL

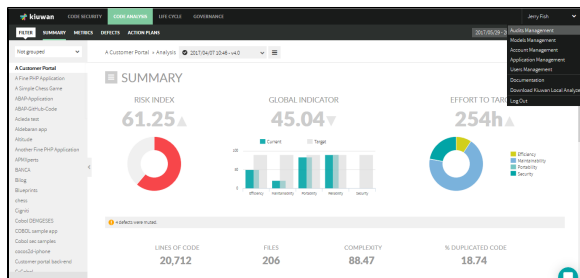
The audit will be **OK**. The mandatory checkpoint is OK and the sum of successful checkpoints (80%) is higher than Audit Approval Threshold (75%).

Audit Management

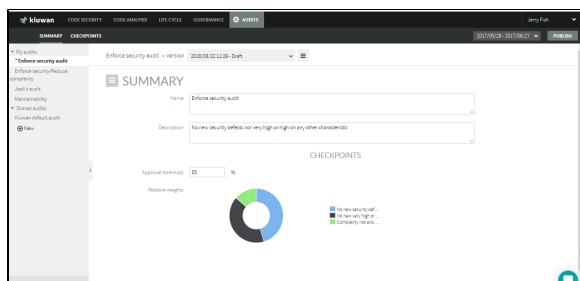
To access the Audit Management module, select **Audits Management** from the configuration drop-down menu.



Only users with "Manage audits" privilege will be allowed to access the Audit module.



You will go directly to the audit summary page for the default selected Audit.

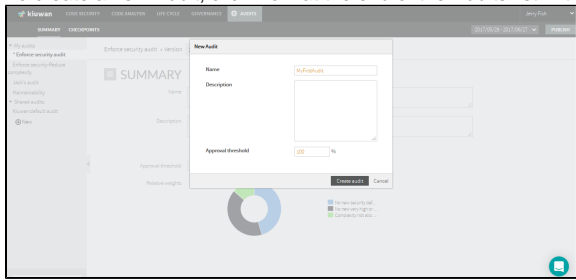


Kiwan comes with an off-the-shelf **Default audit**. This audit cannot be modified by end-users but can be used in any application. This is the Audit assigned to any new application in Kiwan by default.

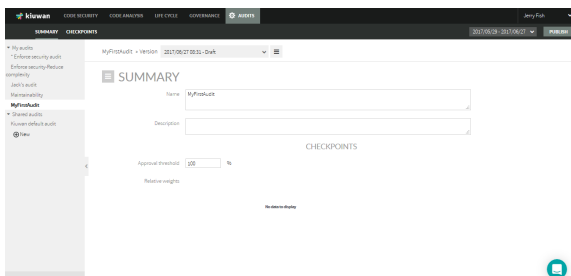
The default audit comes under the **Shared Audits** section in the left panel. Any user-defined audit will be under **My audits**. Clicking on any audit name will allow you to view/manage it.

Create an Audit

1. To create a new Audit, click **New** at the end of the Audits list in the left side panel.



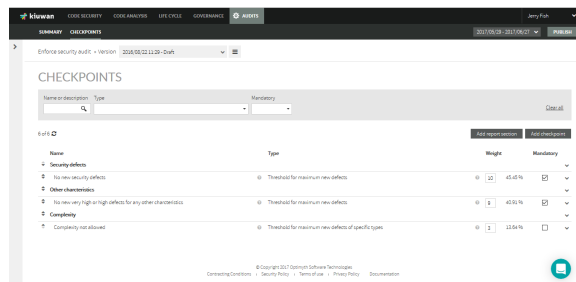
2. Provide a **Name** and (optionally) a **Description**.
3. The **Approval Threshold** represents the minimum percentage of checkpoints contribution to consider the Audit as OK. After audit execution, this value is used to evaluate if the audit passes or not. Please see the [Audit evaluation logic](#) above to know how this value is used in audit evaluation.
4. Click **Create Audit** to save the new audit and have it available under **My audits**.



5. Every Audit needs to have at least 1 Checkpoint. Therefore, once you create an audit, the next step is to create checkpoints.

Checkpoint management

For any selected Audit, the Checkpoints tab will show all the defined checkpoints in a table.



To facilitate working with checkpoints, filter them by Name, Type or Mandatory status in the filter panel above the table.

Click the checkpoint **Name** to modify the checkpoint details and definition.

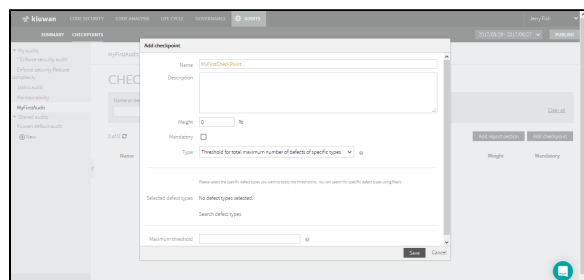
Further actions on this page:

Action	Description
Weight text box	Enter integers in the text box and Kiuwan will automatically calculate the contribution percentage of every checkpoint in the audit. This way, you can easily fine-tune checkpoint contributions without editing every individual checkpoint.
Mandatory	Select the checkbox to make the checkpoint(s) mandatory.
Add Report Section	Click this button to create sections to group checkpoints into sections. You can define the order of sections and checkpoints using the arrows in the first column of the checkpoints table.

To move checkpoints across sections just use the arrows in the checkpoint until you place it in the section you want. The order defined here is used just to display results, it doesn't affect the [Audit evaluation logic explained above](#).

Create a Checkpoint

To create a new checkpoint, click **Add checkpoint**.



Name	Description
Name	Enter a name for the checkpoint (mandatory)
Description	Enter a description of the checkpoint (optional)
Weight	<p>The relative weight of this checkpoint in the Audit.</p> <p>Every checkpoint has an associated Weight that represents the relative weight of the checkpoint in the Audit. The weights you specify (integer) into a percentage contribution to the overall Audit. Please see Audit evaluation logic to fully understand how this value is used in the audit results.</p>
Mandatory	The checkbox indicates if the checkpoint is Mandatory (checked) or Optional (unchecked).
Maximum Threshold	It indicates the maximum number of defects that are allowed. When the audit is executed, if the number of defects is higher than this value, the audit fails.
Type	<p>The drop-down menu allows selecting between the available checkpoint types. Please see Checkpoint Types for an explanation.</p> <div><p>Type</p><p>Very High defects percentage</p><p>Threshold for total maximum number of defects of specific types</p><p>Threshold for maximum new defects of specific types</p><p>Threshold for total maximum number of defects</p><p>Threshold for maximum new defects</p><p>Global indicator improvement</p><p>Global indicator threshold</p><p>Duplicated code threshold</p><p>Very High defects percentage</p></div> <p>Available checkpoint types:</p> <ul style="list-style-type: none">• Threshold for total maximum number of defects of specific rules: sets a maximum number of defects allowed in the application for the specified rules.• Threshold for maximum new defects of specific rules: sets a maximum number of new defects allowed in the application for the specified rules.• Threshold for total maximum number of defects: sets a maximum number of defects allowed in the application for the defined languages, categories and priorities.• Threshold for maximum new defects: sets a maximum number of new defects allowed in the application for the defined languages, categories and priorities.• Global indicator by CQM characteristic threshold: checks if the Global indicator for the specified characteristic is above the defined threshold.• Global indicator improvement: checks if the Global indicator improves the baseline Global indicator.• Global indicator threshold: checks if the Global indicator is above the defined threshold.• Duplicated code threshold: checks if the percentage of Duplicated code is above a defined threshold.• Very High defects percentage: checks if the percentage of Very High defects is above a defined threshold.• Threshold for maximum insights components by severity risk: checks if the number of components that meet the defined severity risk filter is above a defined threshold.• Insights filter components by group, name and version: checks if any of the discovered components meet the defined group, name, version and comparison operator. <p>Depending on the selected checkpoint type, you may be able to specify a filter:</p> <ul style="list-style-type: none">• For defects: only defects of selected languages, characteristics and priorities will be taken into account when evaluating the checkpoint.• For components: only components that match the specified group, name and version, using the defined comparison operator. <p>In the following checkpoint evaluation will only compare with the threshold those defects of Very High priority belonging to Maintainability or Security.</p>

Search defect types

Search and filter types

Name or description	Language	Characteristic	Vulnerability type	Priority	Effort	Tag	Engine	Host	Details
Quick	Java R								Details
<input type="checkbox"/> Only Code Security rules									

1 item selected

Name	Language	Characteristic	Vulnerability type	Priority
<input checked="" type="checkbox"/> Avoid insecure connector products in HTTP and native SQL queries	Java	Efficiency	None	High
<input type="checkbox"/> Use exception method (getter/setter) for property access	Java	Maintainability	None	Low
<input type="checkbox"/> Avoid many-to-many associations	Java	Maintainability	None	Low
<input type="checkbox"/> Avoid raw data type mapping to same database table	Java	Reliability	None	Low
<input checked="" type="checkbox"/> Avoid binding parameter collections	Java	Efficiency	None	Low
<input type="checkbox"/> Avoid setting NullValue that could produce null data rows	Java	Reliability	None	Low
<input type="checkbox"/> Avoid using "0" in mapping operations and in code	Java	Portability	None	Low
<input type="checkbox"/> Avoid using native SQL in mapping operations and in code	Java	Portability	None	Low
<input type="checkbox"/> Use bind or named parameters in HTTP and native SQL queries	Java	Security	Injection	High

Next Cancel

Type

Threshold for total maximum number of defects of specific types

Please select the specific defect types you want to apply the threshold to. You can search for specific defect types using filters.

Selected defect types

Avoid unaware cartesian products in HQL and native SQL queries

Avoid more than one entity mapped to same database table

[Search defect types](#)

Once you have created the audit and its checkpoints, open the [Applications management](#) module, select the **Application** and open the dropdown menu, to associate that audit to an application.

Select the **Audit option** to open a form where you can select available audits.

Selecting an audit will associate that audit to the application, i.e. every delivery analysis on that application will run the audit.

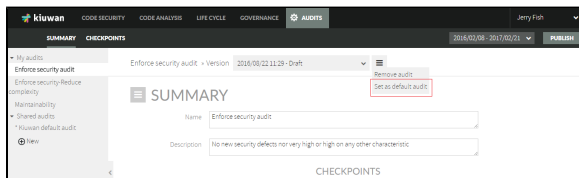
Setting a custom Audit as the default audit for new applications

The **Kiuwan default audit** is the **default** audit assigned to a new application. You can change the audit assigned to an application at any time as seen above.

To set a specific audit, a shared one or one of your custom audits, as the default audit for new applications, do as follows.

Go to **Audits Management** and select the audit you want to be default audit in the audits selector.

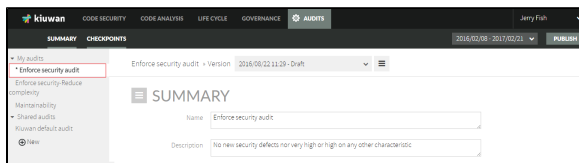
Once selected, open the hamburger menu next to the version number and click **Set as default audit**.



Afterward, the selected audit will be default audit assigned to new applications.

How can I know which is the default audit?

The Audit menu on the left of Audit Management indicates with an asterisk the default audit.



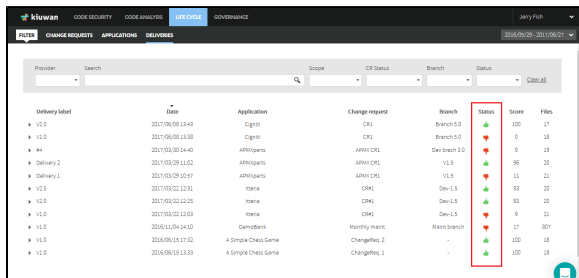
Audit Results

Every time a delivery analysis is executed, the audit associated with that application will be evaluated, associating a value of OK or FAIL to that delivery analysis. You can access the audit results in several ways:

- Interactively, by browsing the Audit Results page
- Programmatically, by inspecting KLA return codes

Audit Results Page

The deliveries module displays a full list of deliveries.

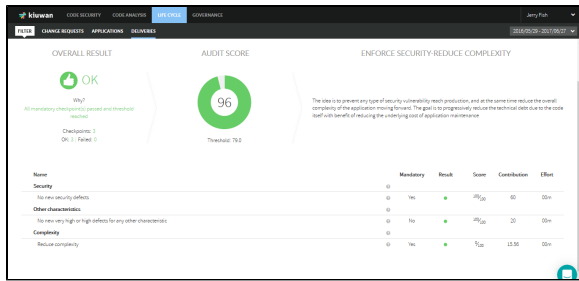


Provider	Search	Scope	CR Status	Branch	Status	Score	Files
Delivery label							
10.0	2017/08/08 13:45	Cignit	CRS	Branch 0.0	▲	100	27
10.0	2017/08/08 13:38	Cignit	CRS	Branch 0.0	▲	0	18
41	2017/08/08 14:40	APMQuarts	APMWS CRS	DevBranch 0.0	▲	9	18
Delivery 2	2017/08/28 11:02	APMQuarts	APMWS CRS	V1.0	▲	96	20
Delivery 1	2017/08/28 09:47	APMQuarts	APMWS CRS	V1.0	▲	11	20
10.0	2017/08/22 12:31	Itaria	CRMS	Dev-1.0	▲	81	20
10.0	2017/08/22 12:29	Itaria	CRMS	Dev-1.0	▲	81	20
10.0	2017/08/22 12:29	Itaria	CRMS	Dev-1.0	▲	9	20
10.0	2018/11/04 14:20	DarioBark	Monthly maint.	Main branch	▲	17	801
10.0	2018/09/10 17:02	A Simple Chess Game	Changefreq 2	-	▲	100	18
10.0	2018/09/10 12:53	A Simple Chess Game	Changefreq 1	-	▲	100	18

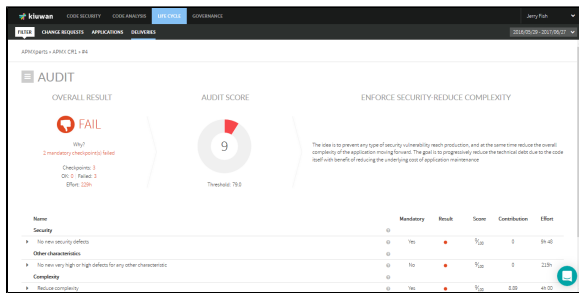
The **Status** column shows the results of the audit as well as the overall compliance of the analysis with the defined audit (**Score** column).

Click the hand icon under the Status column to open the Audit Results page.

The below picture shows an example of an audit that resulted in OK.

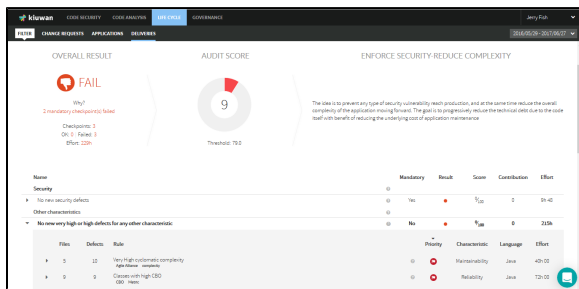


The information is displayed about the **Overall status**, the reasons (**Why?**), the overall compliance (**Audit score**), information on Checkpoint results, the **Effort** needed to pass the audit (if it failed) and specific information about the results of every checkpoint.



A full list of audit's checkpoints is displayed, along with information on each of them.

For those failed checkpoints, click the arrow on the left to list those defects that caused the checkpoint failure.



Click the PDF link to export Audit results in pdf format.

Audit Result when using Local Analyzer

If you are using **Kiuwan Local Analyzer in GUI mode**, after analyzing the delivery, click **View Results** to go directly to the Audit Results page.



In **CLI mode**, **stdout** will display the **URL of the Audit Results Page**

```
Waiting 15 seconds...
Getting A-CDHF Token + a513ce68-3469-4363-a7ce-aa80c6f6174e
Waiting 15 seconds...
Getting A-CDHF Token + 6e00c09f-43ed-400e-a76e-aa00d0d26d00a1
Analysis result URL: https://www.kiuwan.com/saas/web/dashboard/delivery?huu873120DV8W43EAFH52091x31EAV715PNWtubc
E370BP8703918162555GAV6F22
Error: java.lang.NullPointerException
[time analysis: 2100.794 sec]
Shutting down analysis logger...
Task finished successfully.
```

Also, you can check programmatically the audit status (OK or FAIL) by inspecting the return code of KLA script (agent.cmd or agent.sh). Visit [Local Analyzer Return Codes](#) for detailed information on return codes.

Audit Result when using Kiuwan plugin for Jenkins

If you are using the Kiuwan Plugin for Jenkins, you can set the build status depending on audit result, for example by marking the build UNSTABLE in case the audit fails.

Please visit [Jenkins plugin - DeliveryMode](#) for details on how to configure the Kiuwan plugin for Jenkins to set build status depending on audit result.