

User documentation

Blueriq Model Analyzer(BMA) is designed to evaluate Blueriq Models and assess its maintainability and quality. The Model Analyzer is developed to work with the well known code quality platform called SonarQube. The BMA is series of components which either work with or extends SonarQube.

How does it work

When you want to analyze a source code project, being either a Java, C# or Blueriq project, you can leverage SonarQube do handle the analysis for you. SonarQube consist out a few concepts.

SonarQube Server

The main application of SonarQube, where your analysis projects are stored and can be viewed. It also houses registered SonarQube Plugins.

SonarQube Plugin

A plugin is an extension for SonarQube, which can analyse source code. SonarQube delivers plugins for a variety of language, such as Java and C#. These plugins perform the analysis of your source code and sends the results of the analysis to the SonarQube Server.

SonarQube Scanner

A scanner is a tool to execute your source code analysis. SonarQube delivers scanners for a variety of build systems, such as Maven and MSBuild. You configure a scanner to execute your source code analysis. The scanner will fetch all registered plugins from the SonarQube Server and will give the configured source code to every plugin so that the plugin can runs its analysis.

SonarQube Quality Gate

See [Quality gate](#).

Blueriq Model Analyzer

The Model Analyzer consists out of three components. A Plugin, a Scanner and a Quality Gate. The SonarQube Scanners index the gives source code directories and files, so it can run its analysis on those indexed files. Since Blueriq Models live in Blueriq Studio and does not consist out files, we needed to create our own scanner. The scanner creates directories and files in a structure way which represents your Blueriq project structure including repository and branch. Once this file structure is in place, the scanner works just like every other SonarQube Scanner.

Scope of the analysis

The Model Analyzer can only analyze the content of project which are created in the project itself. For more information see [Analysis Scope](#).

Configure analysis

The Blueriq Sonar Scanner needs to be configured to read your Blueriq Model on which the analysis will be run. Blueriq Sonar Scanner takes care of this by reading a application.properties file. See [Configuring Sonar Scanner](#) on how to setup your application.properties file.

Running analysis

To perform an analysis you need to have the Blueriq Sonar Scanner somewhere on you local environment. This Blueriq Sonar Scanner can be found in the zip of the Blueriq Model Analyzer and is called *bma-sonar-scanner.jar*. To perform an analysis you will need to execute a command from a command prompt. The command consist out of two parts. The first part being the execution of java with the Blueriq Sonar Scanner and the second part is the location of the application.properties.

Example	Explanation
java -jar <location on disk>\bma-sonar-scanner.jar	The <location on disk> needs to be replaced with the location of where the bma-sonar-scanner.jar can be found.
-Dspring.config.location=file:<location on disk>\application.properties	The <location on disk> needs to be replaced with location of where the application.properties can be found.

When performing the following command **java -Dspring.config.location=file:.\application.properties -jar C:\tools\bma-sonar-scanner.jar** . The Blueriq Sonar Scanner which is located in **C:\tools** will be executed with application.properties from the **current directory**. When this command is executed, Blueriq Sonar Scanner will either download a branch export from the management service or from a configured location, and perform the analysis. When the analysis is done the command prompt will inform that the analysis was successful.

- [How does it work](#)
 - [SonarQube Server](#)
 - [SonarQube Plugin](#)
 - [SonarQube Scanner](#)
 - [SonarQube Quality Gate](#)
- [Blueriq Model Analyzer](#)
 - [Scope of the analysis](#)
 - [Configure analysis](#)
 - [Running analysis](#)
- [Evaluating analysis results](#)
 - [Code](#)
 - [Measures](#)
 - [Module measures](#)
 - [Rules](#)

```

[INFO] ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/
[INFO] Note that you will be able to access the updated dashboard
[INFO] More about the report processing at http://localhost:9000/
[INFO] Task total time: 30.917 s
[INFO] -----
[INFO] BUILD SUCCESS

```

All the required properties must be set, otherwise the command prompt will show which properties aren't set.

```

Property: blueriq.scanner.workingDirectory
Value: null
Reason: must not be empty

Property: blueriq.scanner.branch
Value: null
Reason: must not be empty

Property: blueriq.scanner.projects
Value: []
Reason: must not be empty

Property: blueriq.scanner.repository
Value: null
Reason: must not be empty

Property: blueriq.scanner.scanName
Value: null
Reason: must not be empty

```

Evaluating analysis results

When the analysis has finished the results can be viewed through a web browser of your choice, at http://<your-machine-name>:<port_number> for instance <http://localhost:9000/>

Analyzed projects are shown under Projects and are clickable to acquire more in depth insight:

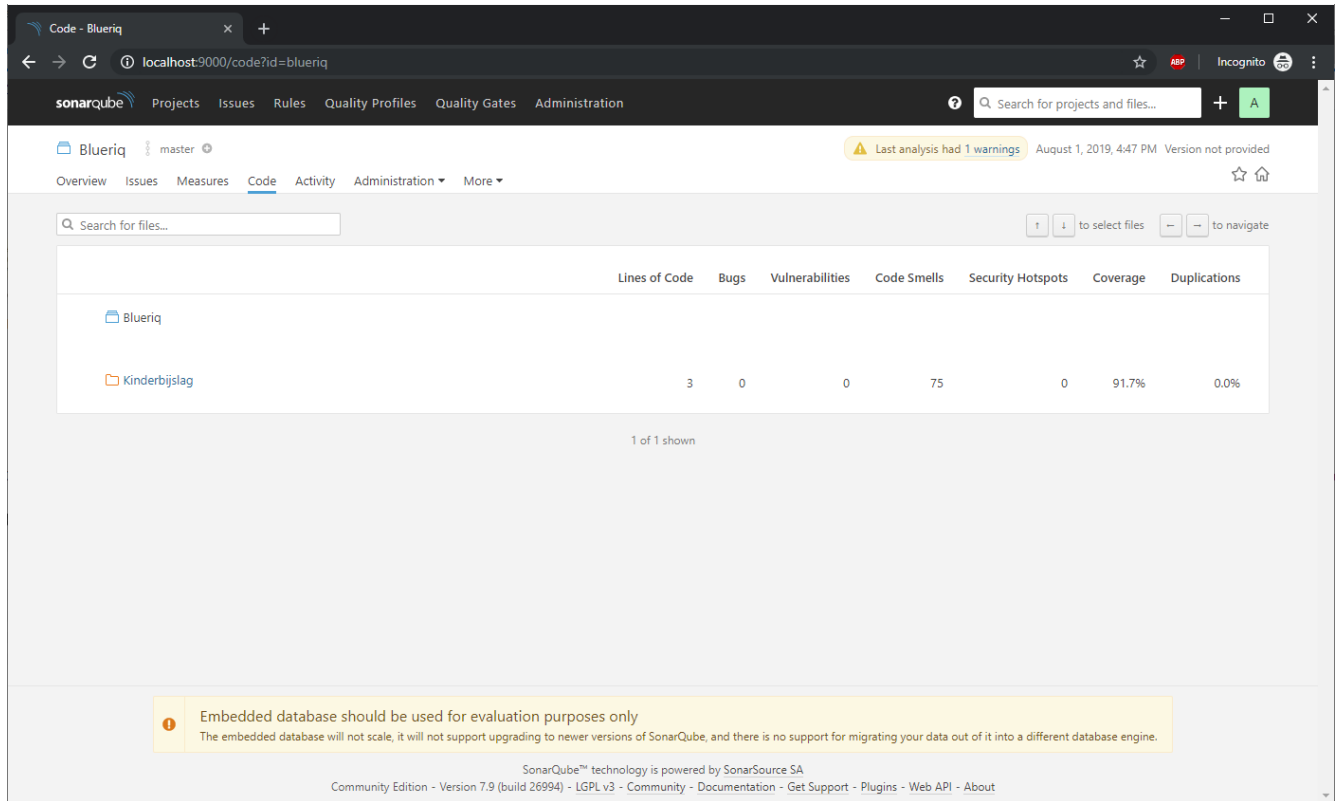
The screenshot shows the SonarQube web interface. The top navigation bar includes 'sonarqube', 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'. A search bar is on the right. The 'Projects' tab is active, showing a list of projects. The 'Blueriq' project is selected, showing a 'Failed' status. The main dashboard for the project displays various metrics: 0 Bugs (green A), 0 Vulnerabilities (green A), 75 Code Smells (red E), 91.7% Coverage (green circle), and 0.0% Duplications (green circle). The last analysis was on August 1, 2019, at 4:47 PM. The left sidebar shows filters for Quality Gate (1 Failed), Reliability (1 A), Security (1 A), and Maintainability (1 A). A yellow warning banner at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

Clicking on the project will show a new screen with three tabs to browse through the analysis results:

- Issues
- Measures
- Code

Code

The third tab in the upper left corner, called Code, gives you an overview of your projects and its quality.



The screenshot shows the SonarQube web interface in the 'Code' tab. The breadcrumb navigation shows 'Blueriq' > 'master'. A table lists the projects. The 'Blueriq' project is selected, and its details are shown in the table below. The table has columns for Lines of Code, Bugs, Vulnerabilities, Code Smells, Security Hotspots, Coverage, and Duplications.

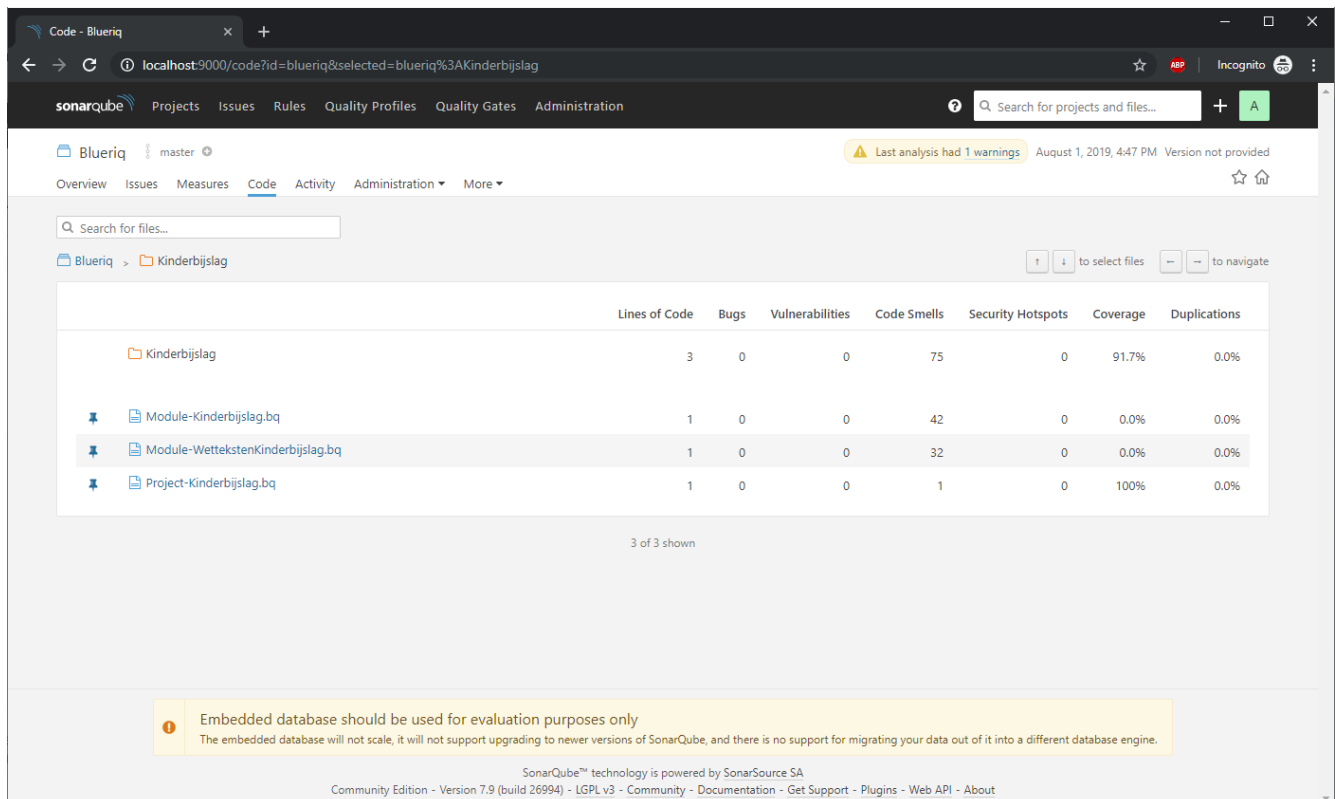
	Lines of Code	Bugs	Vulnerabilities	Code Smells	Security Hotspots	Coverage	Duplications
Blueriq	3	0	0	75	0	91.7%	0.0%

1 of 1 shown

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 7.9 (build 26994) - LGPL v3 - Community - Documentation - Get Support - Plugins - Web API - About

Selecting a project will go one level deeper and display the code smells that are detected on specific modules.



The screenshot shows the SonarQube web interface in the 'Code' tab, now showing the details of the 'Blueriq' project. The breadcrumb navigation shows 'Blueriq' > 'Kinderbijslag'. A table lists the modules. The table has columns for Lines of Code, Bugs, Vulnerabilities, Code Smells, Security Hotspots, Coverage, and Duplications.

	Lines of Code	Bugs	Vulnerabilities	Code Smells	Security Hotspots	Coverage	Duplications
Kinderbijslag	3	0	0	75	0	91.7%	0.0%
Module-Kinderbijslag.bq	1	0	0	42	0	0.0%	0.0%
Module-WettekstenKinderbijslag.bq	1	0	0	32	0	0.0%	0.0%
Project-Kinderbijslag.bq	1	0	0	1	0	100%	0.0%

3 of 3 shown

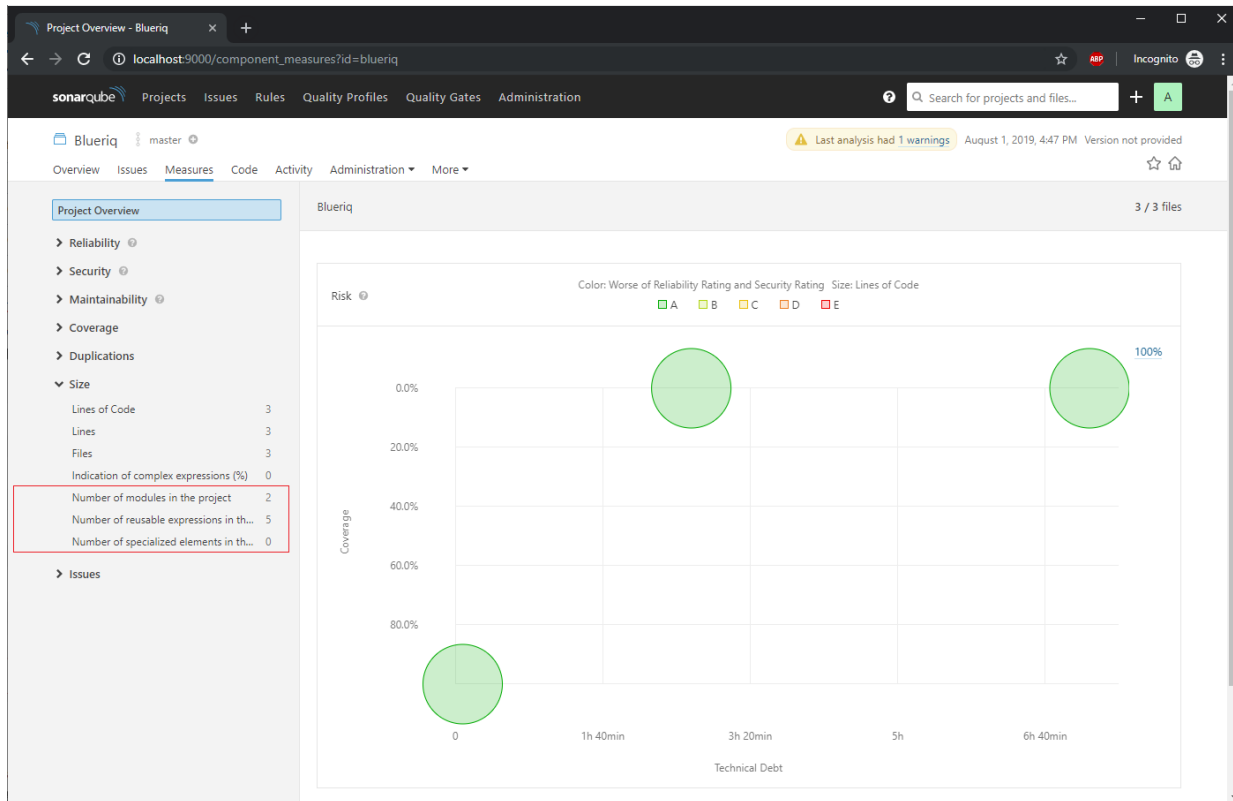
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 7.9 (build 26994) - LGPL v3 - Community - Documentation - Get Support - Plugins - Web API - About

Measures

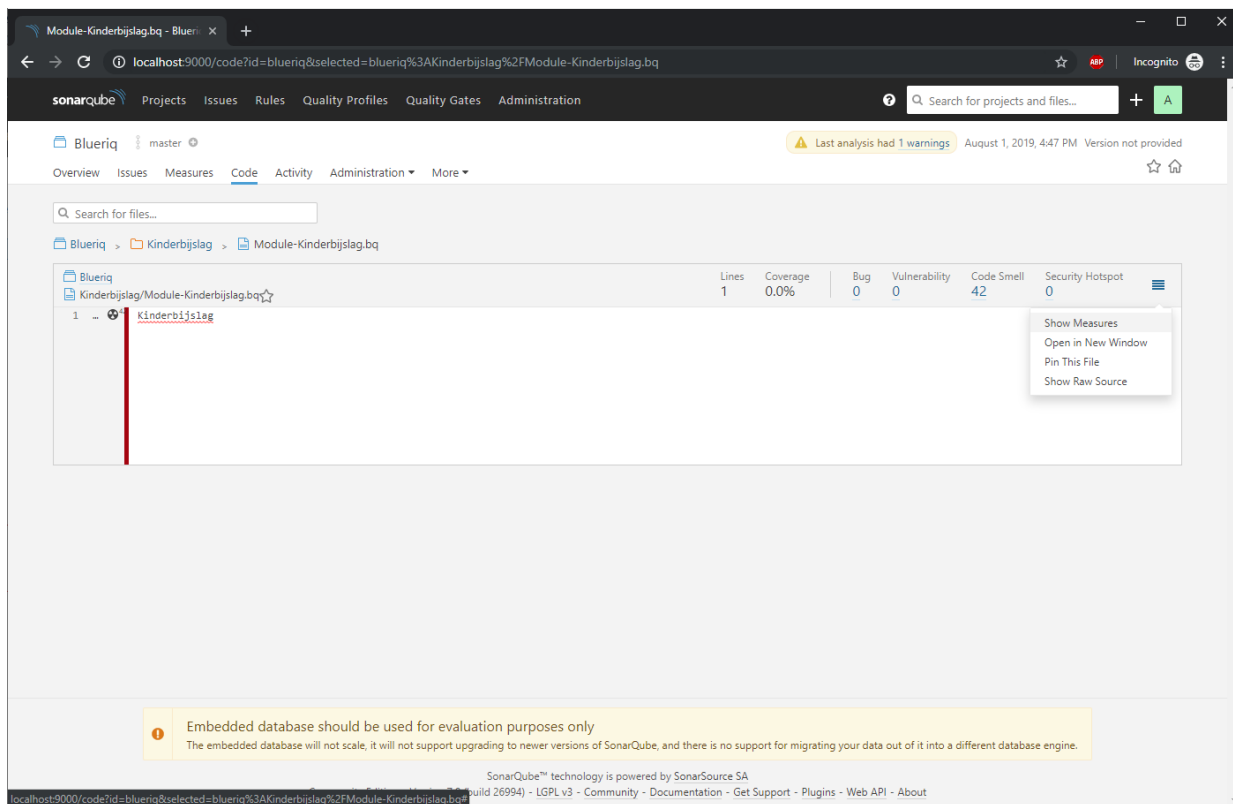
The Blueriq Model Analyzer is capable of calculating the following measures:

You can view the measures of your Blueriq Model by clicking the Measures tab in the upper left corner of your screen, calculated measures are shown in sidebar under the section size:



Module measures

The BMA indexes projects and modules as files in SonarQube. If you want to see module specific measures you will need to open a file in SonarQube and click on show measures.

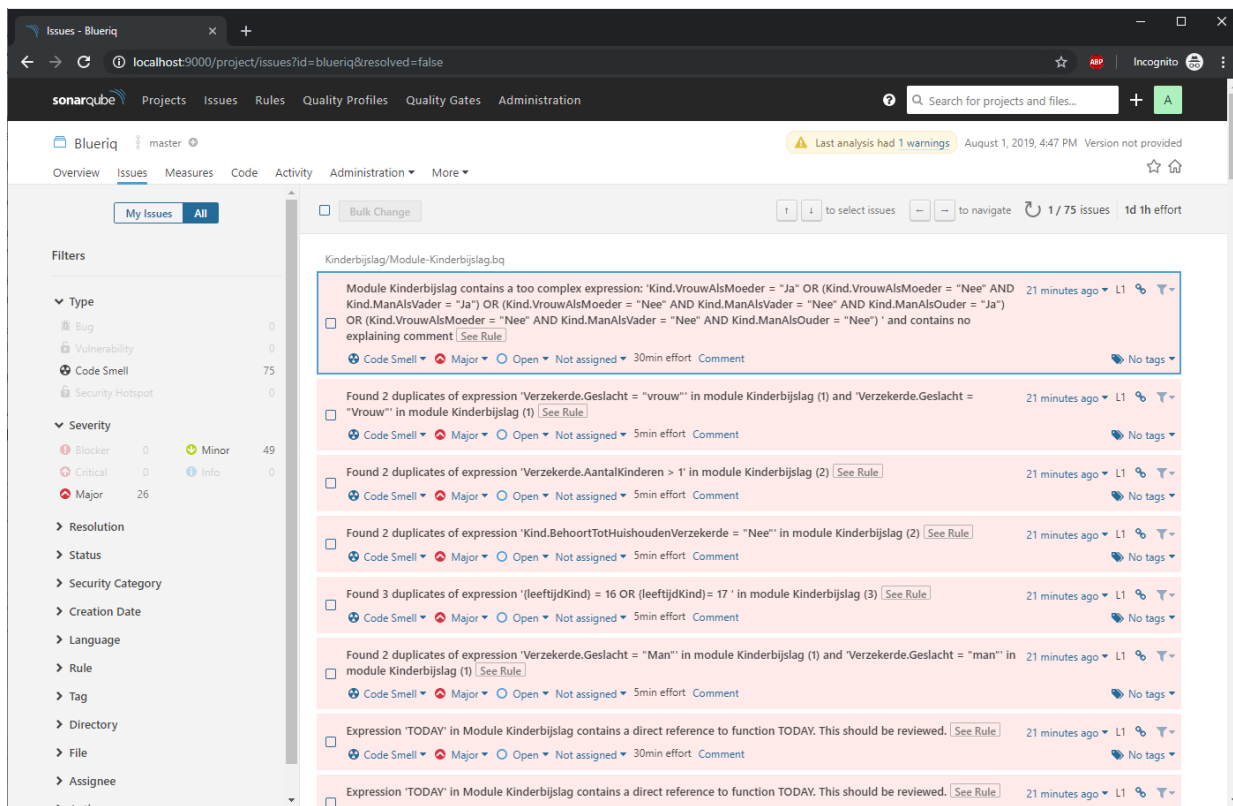


Rules

The Blueriq Model Analyzer is capable of generating model smells based on the following rules:

BMA rule: Aggregates should not contain too many metadata elements	BMA rule: Elements should not contain username and password	BMA rule: Flows should not contain too many elements
BMA rule: AQ_RequestParameters service should be implemented in certain way	BMA rule: Elements should not have too many specializations	BMA rule: Incorrect usage of today
BMA rule: Condition node should have a label	BMA rule: Exposed flows should have role(s) assigned	BMA rule: Library elements should be used
BMA rule: Decision table gap and overlap analysis	BMA rule: Expression is too complex	BMA rule: Module should not contain specialized elements
BMA rule: Decision table should not be used as an attribute default value	BMA rule: Expression may be simplified	BMA rule: Precondition should not evaluate to a constant value
BMA rule: Duplicate expressions should be eliminated	BMA rule: Flow contains incorrect condition node	BMA rule: Projects should not contain diamond shaped module structure
BMA rule: Element names should comply with a naming convention	BMA rule: Flows should fit the default viewport	BMA rule: Reusable expressions should not be duplicated
BMA rule: Elements should not contain connection details	BMA rule: Flows should not contain many complex elements	

The model smells of your Blueriq Model can be viewed by clicking the Issues tab in the upper left corner of your screen:



FAQ

I get the following Exception when running the BMA `javax.xml.bind.UnmarshalException: Unable to create an instance of ...`

- This issue means that you performing an analysis with a branch export which could either be a newer or older version than that is supported by the BMA release you are using.