

Organization Dashboard (Debt Clock)

User Manual — Walkthrough Guide

Step-by-Step Guide with Screenshot Placeholders

Table of Contents

1. Introduction	4
2. Logging In.....	5
3. Main Dashboard	7
3.1 Summary Cards	7
3.2 Cumulative Debt Chart	8
3.3 Project List.....	9
3.4 Developer Rankings	10
Cleaner Contributors	10
Most Refactoring Contributors	10
4. Adding a New Project	12
5. Running an Analysis (Analyze)	13
6. Detailed Project View (More)	14
6.1 Project Metric Cards	15
6.2 Debt / Interest Rate Chart.....	16
6.3 File Metrics Table (Metrics)	17
7. High TD Risk Analysis.....	19
7.1 Class Heatmap.....	20
7.2 File Interpretation	21
7.3 Root Cause Analysis.....	21
Root Causes (Left card)	21
Metrics Threshold (Right card).....	22
8. Refactoring Urgency Predictions.....	23
8.1 Refactoring Probability Heatmap	24
8.2 File Prediction Details	25
8.3 Accepting or Rejecting a Prediction.....	26
8.4 Expert's Opinion	27
Similar Past Cases Table (Left).....	27
Neighborhood Plot (Right)	29
Saving Your Opinion	29
8.5 Retraining the Model.....	30
9. Chart Interactions.....	31

10. Metrics Glossary.....	32
Key Terms:	32
Appendix: Screenshot Checklist.....	33

1. Introduction

The Organization Dashboard (Debt Clock) is a web application for monitoring and analyzing Technical Debt in Java projects. The application analyzes source code, computes quality metrics, detects refactorings, and uses machine learning to predict which files urgently need refactoring.

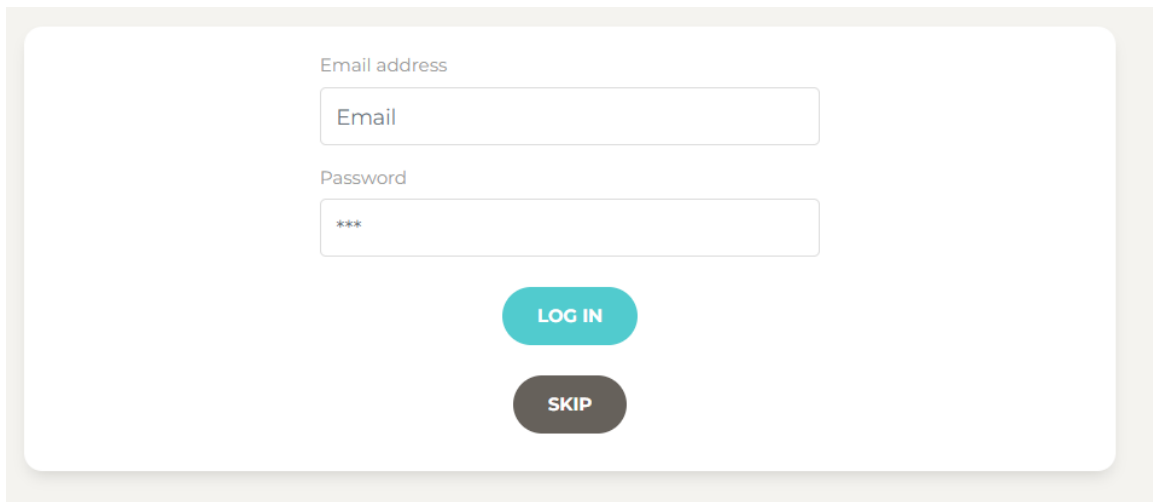
This manual guides you step by step through every feature of the application. At each step there is a screenshot placeholder, insert the corresponding screenshots at the marked positions.

The application is accessible via browser at: <https://app.deptanalyzer.gr/>

2. Logging In

When you first access the application, the Login page is displayed.

[SCREENSHOT S01]

A screenshot of a login page. It features a white rounded rectangle centered on a light beige background. Inside the rectangle, there are two input fields. The first is labeled "Email address" and contains the placeholder text "Email". The second is labeled "Password" and contains the placeholder text "***". Below these fields are two buttons: a teal rounded button labeled "LOG IN" and a dark grey rounded button labeled "SKIP".

Step 1: Open your browser and navigate to <https://app.deptanalyzer.gr/>

→ The Login page appears with a centered card containing the login form.

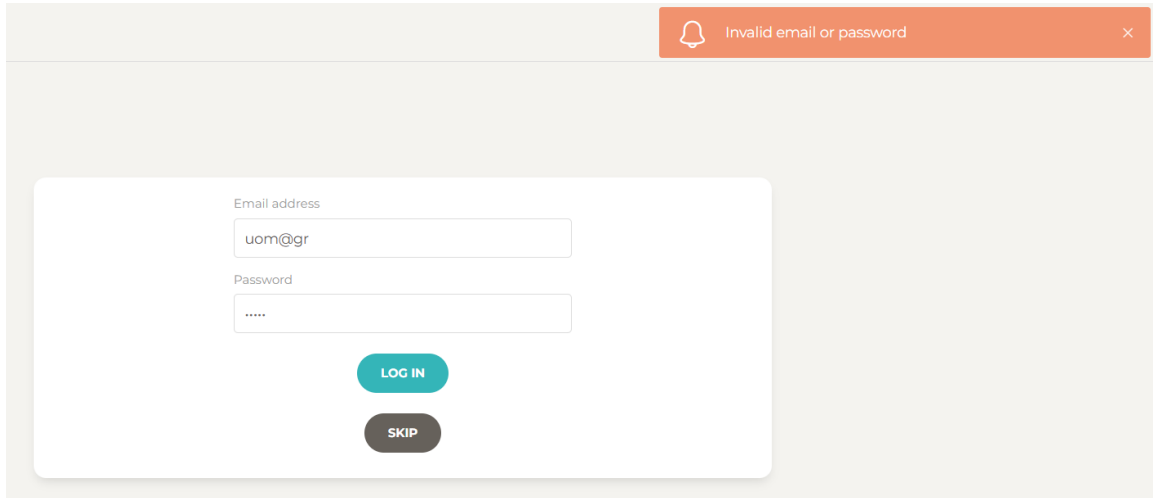
Step 2: In the "Email" field, type your email address (e.g. user@example.com).

Step 3: In the "Password" field (placeholder "***"), type your password.

Step 4: Click the "Log In" button (blue, rounded).

→ If the credentials are correct, you are automatically redirected to the Main Dashboard. If the credentials are wrong, a red notification appears: "Invalid email or password".

[SCREENSHOT S02]



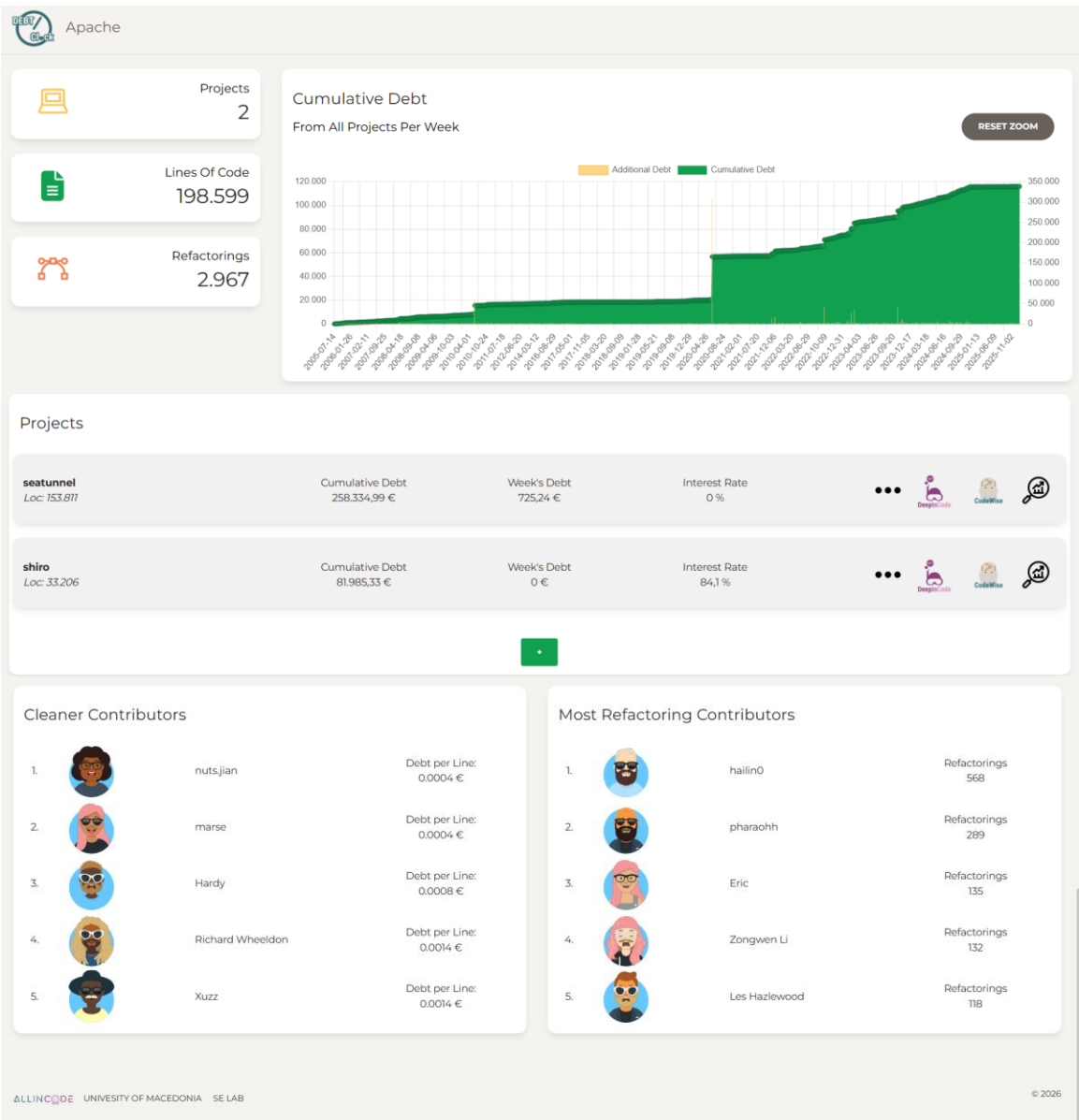
The screenshot shows a login interface. At the top right, an orange notification bar contains a bell icon, the text "Invalid email or password", and a close button (X). Below this, a white login card is centered. It has two input fields: "Email address" with the text "uom@gr" and "Password" with masked characters ".....". Below the fields are two buttons: a teal "LOG IN" button and a gray "SKIP" button.

NOTE: Alternatively, you can click the "Skip" button (gray) to enter without logging in. In this case you will only have read access (GET requests). Operations requiring authentication (e.g. analyze, add project) will not be available.

3. Main Dashboard

After logging in, you are presented with the Main Dashboard — the primary screen of the Organization Dashboard. Here you see a summary of all projects in your organization.

[SCREENSHOT S03]

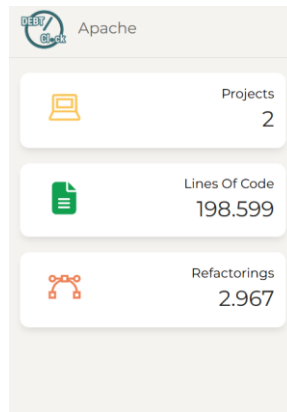


3.1 Summary Cards

In the top-left area there are three summary cards:

Card	Icon	What It Shows
Projects	Globe icon	Total number of projects in the organization
Lines of Code	Chart-pie icon	Total lines of code across all projects
Refactorings	Diamond icon	Total number of detected refactoring operations

[SCREENSHOT S04]



3.2 Cumulative Debt Chart

On the right side, above the project list, there is the main chart titled "Cumulative Debt — From all projects per week".

The chart contains:

- Yellow bars (Additional Debt): New debt added each week (left Y-axis)
- Green line (Cumulative Debt): Accumulated debt in euros (right Y-axis)
- The X-axis shows dates (weekly)

[SCREENSHOT S05]

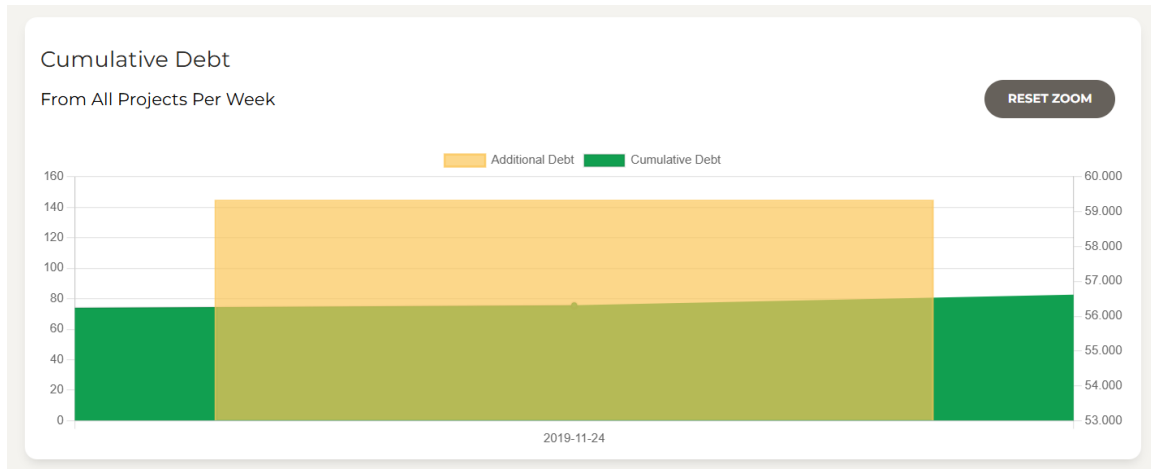


Chart interactions:

Step 1: To Zoom In: Scroll the mouse wheel up over the chart.

→ Focuses on a narrower time range.

Step 2: To Zoom Out: Scroll the mouse wheel down.

→ Shows a wider time range.

Step 3: To Pan: Hold left click and drag.

→ Moves the visible range left or right.

Step 4: To Reset: Click the "Reset Zoom" button (top-right of the chart).

→ Returns to the original view.

NOTE: You can show/hide data series by clicking on the legend labels below the chart.

3.3 Project List

Below the chart, in the "Projects" section, all tracked projects are listed. Each project row/card contains:

Information	Description
Project Name	In bold on the left side
Loc: XX,XXX	Lines of Code (in italics)
Cumulative Debt	Total accumulated technical debt in € (green)

	label)
Week's Debt	Debt added in the most recent week in € (yellow label)
Interest Rate	Rate of debt growth in % (pink label)

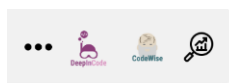
[SCREENSHOT S06]

seatunnel Loc: 153.811	Cumulative Debt 258,334,99 €	Week's Debt 725,24 €	Interest Rate 0 %	...			
---------------------------	---------------------------------	-------------------------	----------------------	-----	---	---	---

On the right side of each row there are 4 action buttons:

Button	Icon	Tooltip	Action
1st	Magnifying glass	"More"	Opens the detailed project view
2nd	DeepInCode logo	"High TD Risk"	Opens the high-risk TD analysis
3rd	CodeWise logo	"Refactoring Urgency"	Opens refactoring predictions
4th	Analysis icon	"Analyze"	Triggers a new code analysis

[SCREENSHOT S07]



3.4 Developer Rankings

At the bottom of the Dashboard there are two ranking cards:

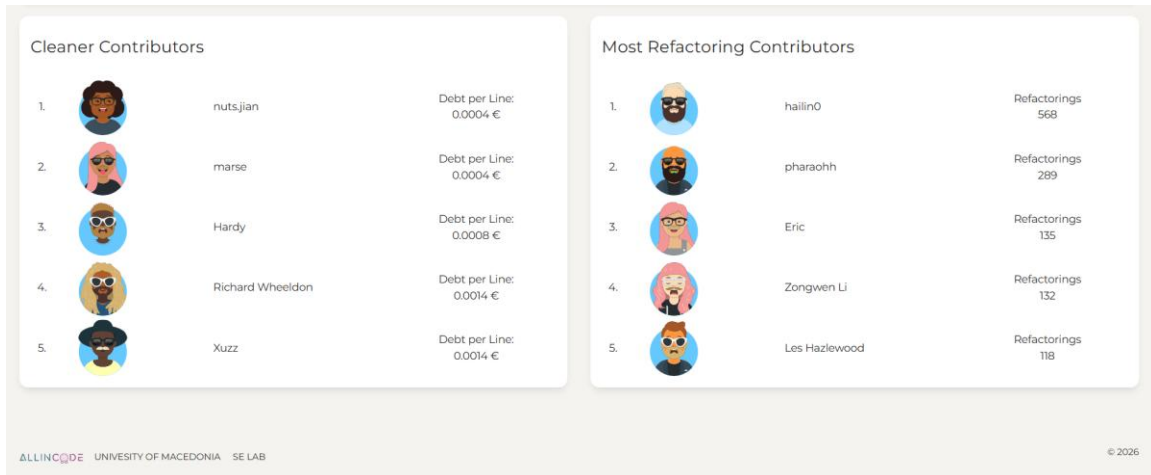
Cleaner Contributors

Ranks developers by lowest Debt per Line of Code — those who write the "cleanest" code. Each row shows: ranking number, avatar, name, and "Debt per Line: X.XX €".

Most Refactoring Contributors

Ranks developers by most refactorings performed. Each row shows: ranking number, avatar, name, and "Refactorings: XX".

[SCREENSHOT S08]



4. Adding a New Project

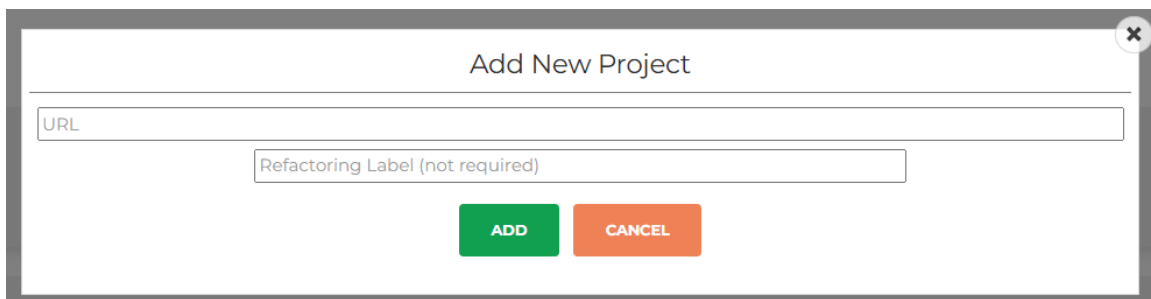
To add a new GitHub repository to your organization, follow these steps:

Step 1: On the Dashboard, scroll down to the "Projects" section.

Step 2: Click the green "+" button (Add Project) located below the project list.

→ A pop-up dialog (modal) appears with the title "Add New Project".

[SCREENSHOT S09]



The screenshot shows a modal dialog titled "Add New Project" with a close button (X) in the top right corner. Inside the dialog, there are two input fields: "URL" and "Refactoring Label (not required)". Below the input fields are two buttons: a green "ADD" button and an orange "CANCEL" button.

Step 3: In the "URL" field, type the GitHub repository URL (e.g. <https://github.com/organization/my-java-project>).

Step 4: (Optional) In the "Refactoring Label (not required)" field, you can type a refactoring label.

Step 5: Click the green "Add" button.

→ The modal closes and the new project appears in the list. You now need to run an Analysis to compute its metrics.

Step 6: Alternatively, click "Cancel" (orange button) to abort.

NOTE: If you leave the URL field empty and click "Add", a validation message appears: "Please fill in all fields."

NOTE: Only Java-based GitHub repositories are supported.

5. Running an Analysis (Analyze)

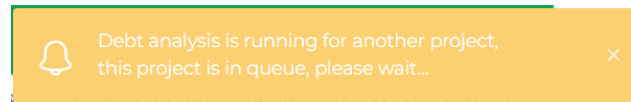
The analysis clones the repository, analyzes each commit, computes code quality metrics, detects refactorings, and calculates technical debt.

Step 1: On the Dashboard, find the project you want to analyze.

Step 2: Click the "Analyze" button (4th button on the right, tooltip "Analyze").

→ The analysis starts. A yellow notification appears in the top-right corner.

[SCREENSHOT S10]



During analysis:

- If you click "Analyze" again on the same project: you see "Debt analysis is running for this project, please wait XX%" with the completion percentage.
- If you click "Analyze" on a different project: you see "Debt analysis is running for another project, this project is in queue, please wait..."
- Only one project is analyzed at a time. Others are queued.

Step 3: Wait for the analysis to complete.

→ After completion, the project metrics are automatically updated on the card.

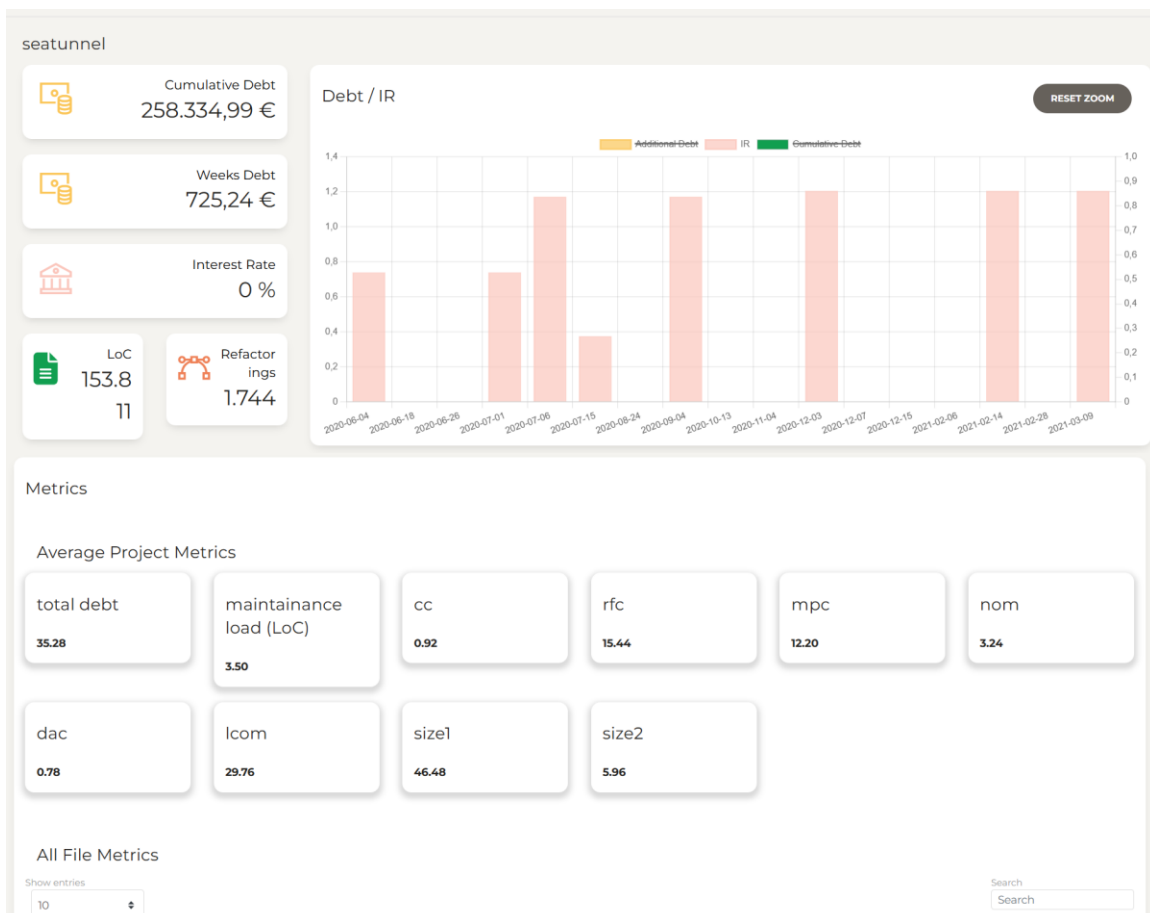
NOTE: Analysis time depends on the project size (number of commits and files). Small projects take a few minutes; large ones may take considerably longer.

6. Detailed Project View (More)

Step 1: On the Dashboard, click the "More" button (magnifying glass, 1st button) on the project you are interested in.

→ You are taken to the detailed project view (Project More). The project name is displayed at the top.

[SCREENSHOT S11]



FILE	TOTAL DEBT	MAINTAINANCE LOAD (LOC)	CC	RFC	MPC	NOM	DAC	LCOM	SIZE1	SE
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/TypesafeConfigUtils.java	13.025059984572826	0	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/ConfigRuntimeException.java			-1	0	0	0	0	0	8	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/Common.java	1668.5729380423775	2	1.5714285373687744	91	77	14	1	91	92	21
seatunnel-common/src/main/java/org/apache/seatunnel/common/PropertiesUtil.java	13.882879999999998	2	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/CheckResult.java	10.569919985895156	0	1	2	0	2	1	1	7	5
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/CheckConfigUtil.java	19.24672	0	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/Constants.java	54.900479999999999	0	-1	0	0	0	0	0	9	14
seatunnel-common/src/test/java/org/apache/seatunnel/common/config/CheckConfigUtilTest.java	18.1846059471532	2	0.800000011920929	12	8	4	0	6	47	4
seatunnel-common/src/test/java/org/apache/seatunnel/common/config/TypesafeConfigUtilsTest.java	29.903783603218848	0	0.75	9	6	3	0	3	25	3
seatunnel-config/seatunnel-config-shade/src/main/java/org/apache/seatunnel/shade/com/typesafe/config/impl/ConfigParser.java	0.31551999999999997	1	0	0	0	0	0	0	0	0
FILE	TOTAL DEBT	MAINTAINANCE	CC	RFC	MPC	NOM	DAC	LCOM	SIZE1	SE
Showing 1 to 10 of 3309 entries										
Previous 1 2 3 4 5 6 7 8 Next										

Cleaner Contributors

1.  nuts.jian Debt per Line: 0.0004 €

2.  marse Debt per Line: 0.0004 €

3.  Hardy Debt per Line: 0.0008 €

4.  Xuzz Debt per Line: 0.0014 €

5.  chenqianwen Debt per Line: 0.0019 €

Most Refactoring Contributors

1.  hailin0 Refactorings 568

2.  Eric Refactorings 135

3.  Zongwen Li Refactorings 132

4.  Jia Fan Refactorings 81

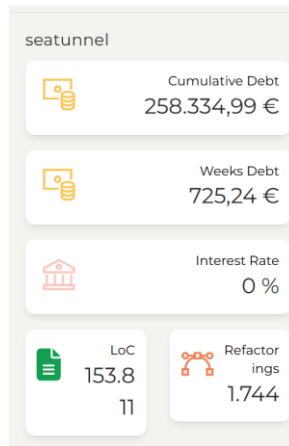
5.  happyboy1024 Refactorings 74

6.1 Project Metric Cards

On the left side there are 5 cards:

Card	Icon	Value
Cumulative Debt	money-coins	XX.XX € — Total accumulated debt
Weeks Debt	money-coins	XX.XX € — Debt from the most recent week
Interest Rate	bank	XX.XX % — Rate of debt growth
LoC	file-lines	XX,XXX — Lines of Code
Refactorings	vector	XX — Detected refactoring operations

[SCREENSHOT S12]

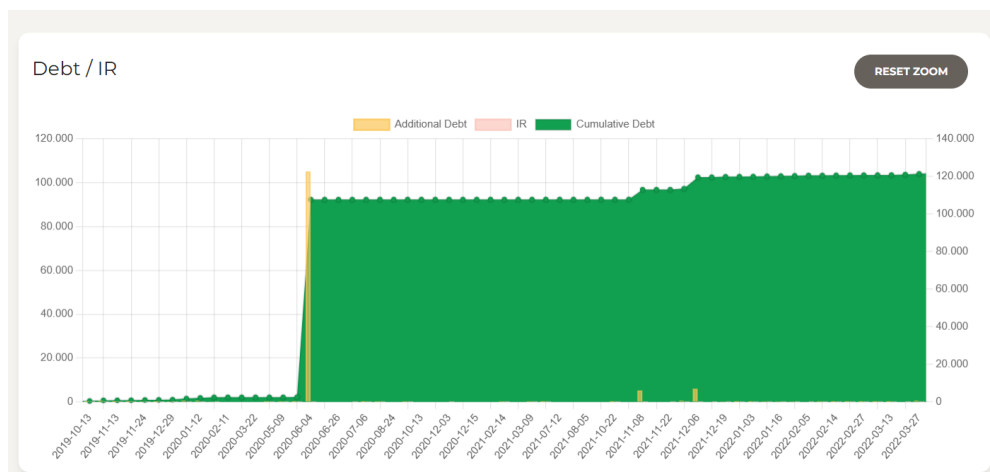


6.2 Debt / Interest Rate Chart

To the right of the cards there is a chart titled "Debt / IR":

- Yellow bars — Additional Debt: New debt per week (hidden by default — click the legend to show)
- Pink bars — IR (Interest Rate): Rate of debt growth per week
- Green line — Cumulative Debt: Accumulated debt (hidden by default)

[SCREENSHOT S13]



NOTE: To show the hidden series (Additional Debt, Cumulative Debt), click on the corresponding label in the legend below the chart.

6.3 File Metrics Table (Metrics)

Below the chart there is the "Metrics" table with per-file metrics:

Column	Description
File	Full file path within the repository
Total Debt	Accumulated debt for this file in €
K	Debt coefficient (K metric)
Complexity	Cyclomatic Complexity
RFC	Response for a Class
MPC	Message Passing Coupling
NOM	Number of Methods
DAC	Data Abstraction Coupling
LCOM	Lack of Cohesion of Methods
SIZE1 / SIZE2	Size metrics

At the bottom of the table there is a row showing the average values for each metric.

[SCREENSHOT S14]

Metrics

Average Project Metrics

total debt

35.28

maintenance
load (LoC)

3.50

CC

0.92

rfc

15.44

mpc

12.20

nom

3.24

dac

0.78

lcom

29.76

size1

46.48

size2

5.96

All File Metrics

[Show entries](#)

10

Search

Search

FILE	TOTAL DEBT	MAINTENANCE LOAD (LOC)	CC	RFC	MPC	NOM	DAC	LCOM	SIZE	SI
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/TypesafeConfigUtils.java	13.025059984572826	0	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/ConfigRuntimeException.java			-1	0	0	0	0	0	8	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/Common.java	1668.5729380423775	2	1.5714285373687744	91	77	14	1	91	92	21
seatunnel-common/src/main/java/org/apache/seatunnel/common/PropertiesUtil.java	13.882879999999998	2	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/CheckResult.java	10.569919985895156	0	1	2	0	2	1	1	7	5
seatunnel-common/src/main/java/org/apache/seatunnel/common/config/CheckConfigUtil.java	19.24672	0	0	0	0	0	0	0	0	0
seatunnel-common/src/main/java/org/apache/seatunnel/common/Constants.java	54.90047999999999	0	-1	0	0	0	0	0	9	14
seatunnel-common/src/test/java/org/apache/seatunnel/common/config/CheckConfigUtilTest.java	18.1846059471532	2	0.800000011920929	12	8	4	0	6	47	4
seatunnel-common/src/test/java/org/apache/seatunnel/common/config/TypesafeConfigUtilsTest.java	29.903783603218848	0	0.75	9	6	3	0	3	25	3
seatunnel-config/seatunnel-config-shade/src/main/java/org/apache/seatunnel/shade/com/typesafe/config/impl/ConfigParser.java	0.3155199999999997	1	0	0	0	0	0	0	0	0
FILE	TOTAL DEBT	MAINTENANCE LOAD (LOC)	CC	RFC	MPC	NOM	DAC	LCOM	SIZE	SI

7. High TD Risk Analysis

The High TD Risk analysis uses a classification model (Machine Learning) to identify files with a high probability of accumulating technical debt.

Step 1: On the Dashboard, click the "High TD Risk" button (2nd button, DeepInCode logo) on the project you are interested in.

→ You are taken to the High TD Risk page. If results already exist, they appear immediately. Otherwise, the analysis runs and a loading spinner is displayed.

[SCREENSHOT S15]



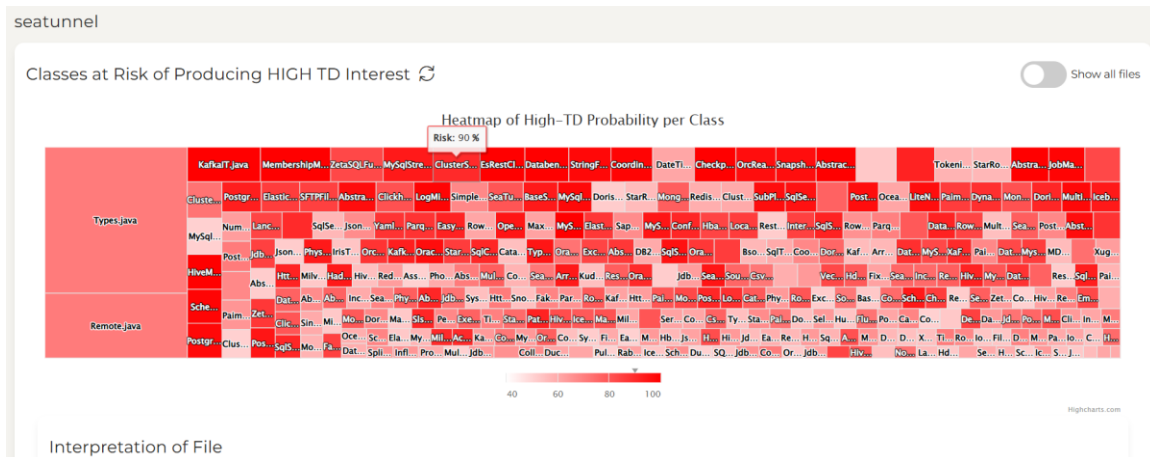
7.1 Class Heatmap

The main visualization is a Treemap Heatmap titled "Heatmap of High-TD Probability per Class".

Each rectangle represents a class/file:

- Rectangle size = Lines of Code (larger = more lines)
- Color: White → Red (redder = higher probability of High-TD)
- Tooltip: Shows "Risk: XX%" on hover

[SCREENSHOT S16]



Available actions:

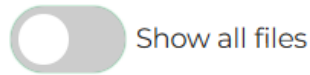
Step 2: To see only high-risk files, make sure the "Show all files" toggle switch is turned off (default state).

→ Only files classified as High-TD are displayed.

Step 3: To see all files, turn on the "Show all files" toggle (click on it).

→ The heatmap updates to show all files in the project.

[SCREENSHOT S17]



Step 4: To refresh results on the latest commit, click the refresh icon (↺) next to the title (tooltip: "Calculate on last commit").

→ The analysis re-runs. A loading spinner appears until completion.

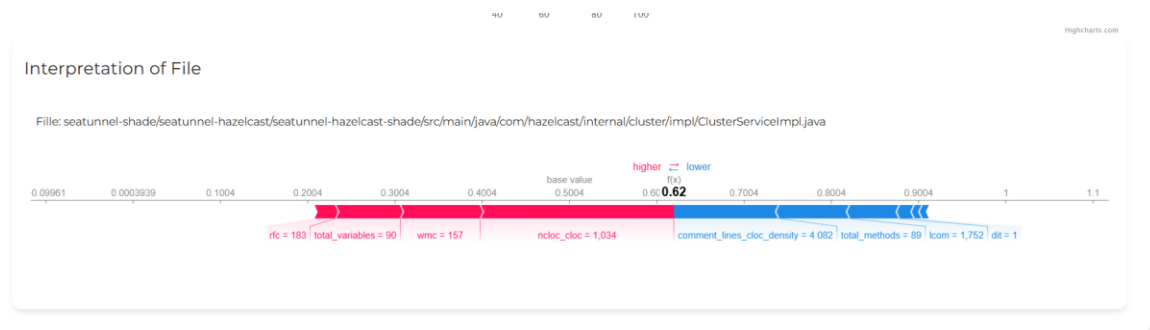
7.2 File Interpretation

Below the heatmap, the "Interpretation of File" section initially shows "Select a file..."

Step 5: Click on a rectangle (class) inside the heatmap.

→ Below the heatmap, the interpretation appears: the file path ("File: path/to/Class.java") and a chart showing which metrics contribute to the risk level.

[SCREENSHOT S18]



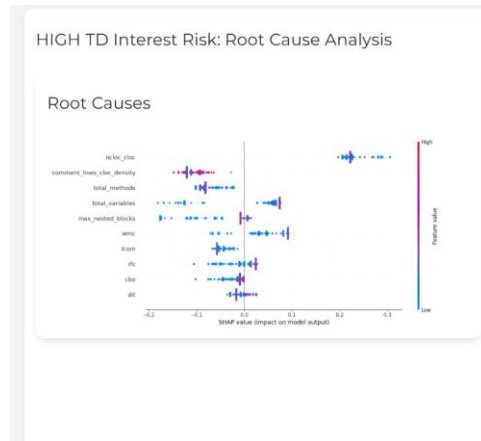
7.3 Root Cause Analysis

At the bottom of the page, the "HIGH TD Interest Risk: Root Cause Analysis" section contains two cards:

Root Causes (Left card)

Displays a bar chart image showing which code quality metrics are the main drivers of high technical debt in the project (e.g. Complexity, CBO, LCOM, etc.).

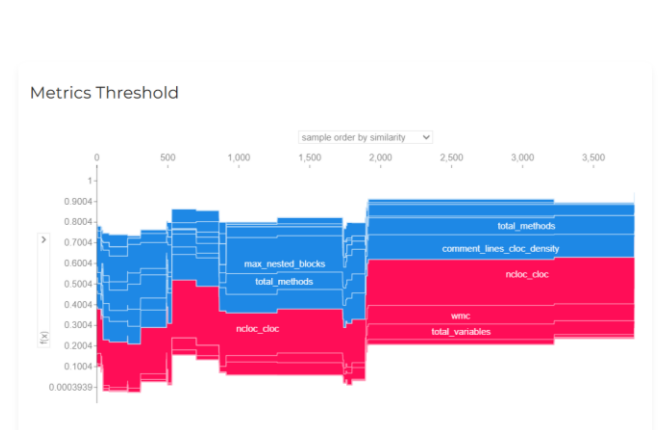
[SCREENSHOT S19]



Metrics Threshold (Right card)

Displays a table with the threshold values for each metric. Files exceeding these thresholds across multiple metrics are classified as high-risk.

[SCREENSHOT S20]



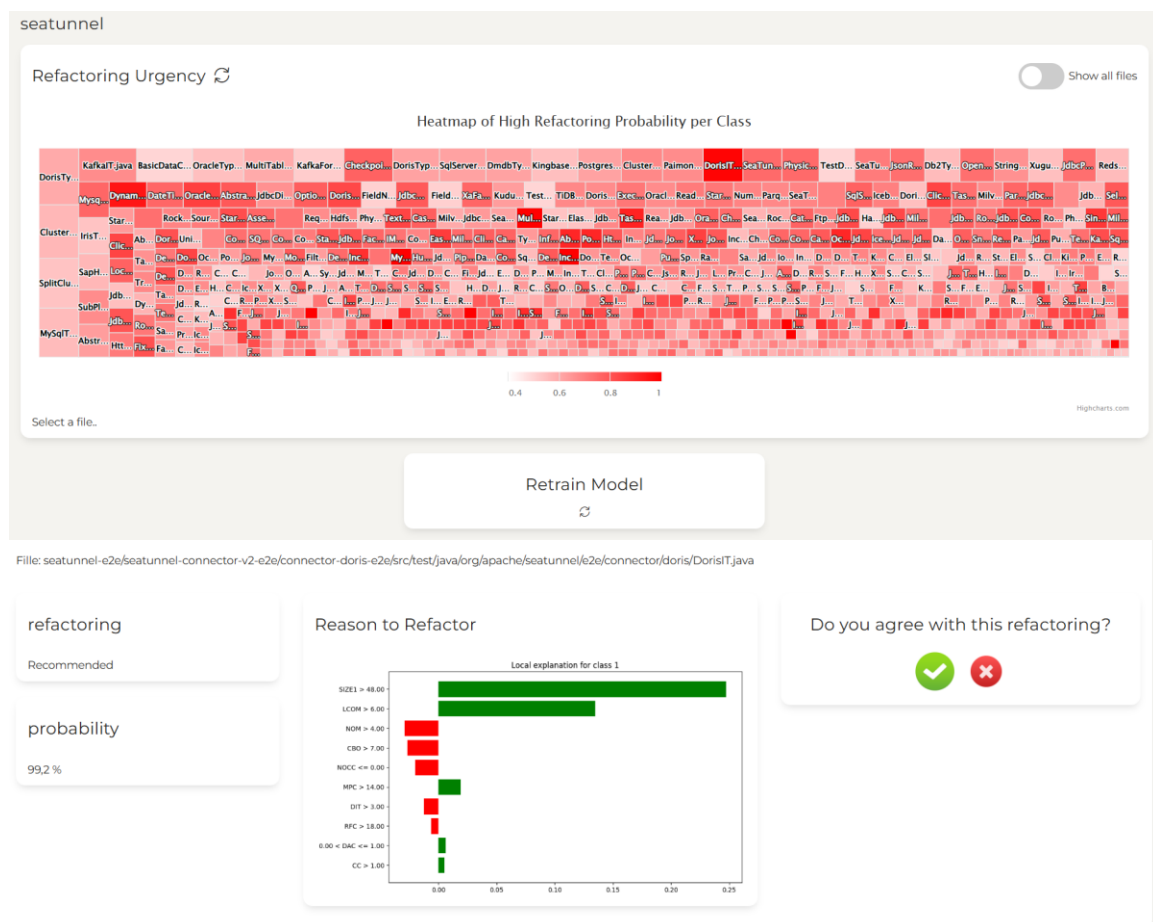
8. Refactoring Urgency Predictions

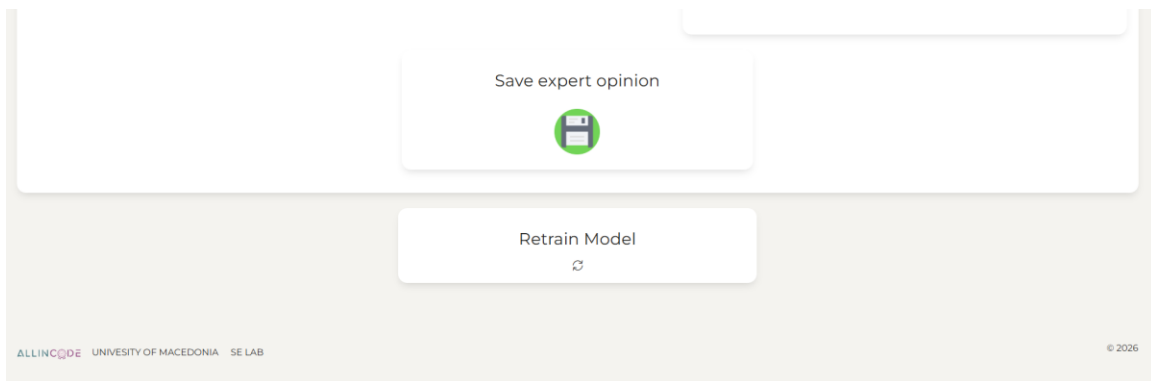
The Refactoring Urgency page uses a machine learning model to predict which files urgently need refactoring. It includes a human-in-the-loop feedback system to improve predictions over time.

Step 1: On the Dashboard, click the "Refactoring Urgency" button (3rd button, CodeWise logo) on the project you are interested in.

→ You are taken to the Refactoring Urgency page. If results already exist, they appear immediately. Otherwise, a loading spinner and "Status" button are shown.

[SCREENSHOT S21]





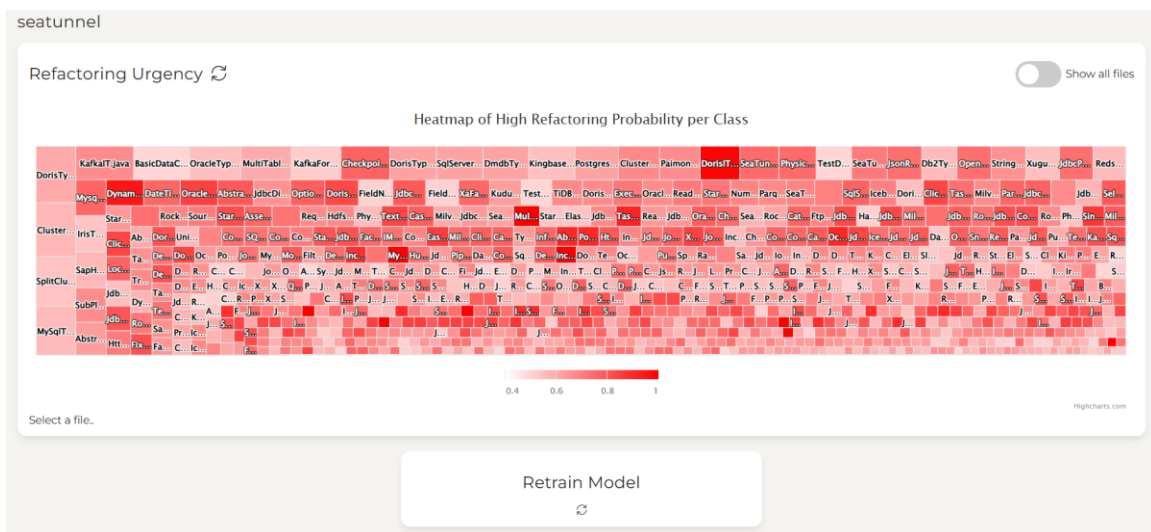
8.1 Refactoring Probability Heatmap

In the center there is a Treemap Heatmap titled "Heatmap of High Refactoring Probability per Class".

Each rectangle:

- Size = Lines of Code
- Color: White → Red (redder = higher probability of needing refactoring)
- Tooltip: "Name: ClassName — Probability: XX"

[SCREENSHOT S22]



Step 2: Toggle the "Show all files" switch to display all files or only those that need refactoring.

Step 3: Click ↻ (refresh, tooltip "Calculate on last commit") to refresh on the latest commit.

8.2 File Prediction Details

Below the heatmap you initially see "Select a file...".

Step 4: Click on a rectangle (class) in the heatmap.

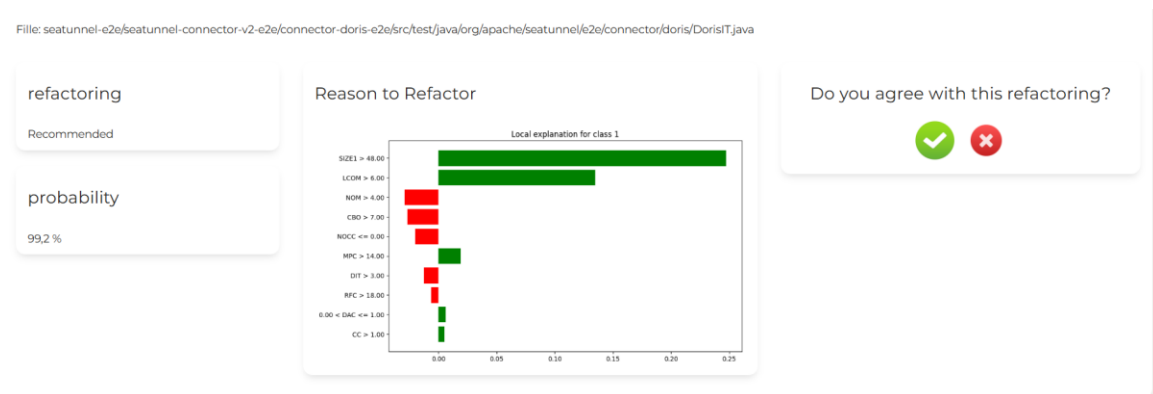
→ Three sections appear:

A) Prediction Metric Cards (left):

Cards in a 3-column grid, each showing a metric and its value. Values can be:

- "Recommended" (green) — Refactoring is recommended for this metric
- "No Need" (gray) — No refactoring needed
- A percentage (e.g. "75%") — Probability of needing refactoring

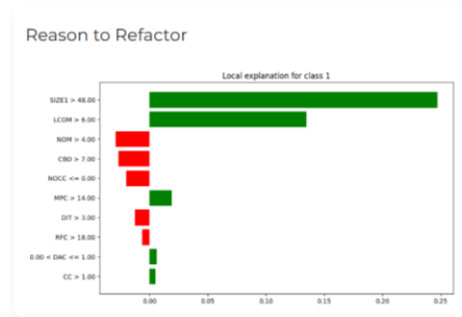
[SCREENSHOT S23]



B) Reason to Refactor (center):

An image showing graphically which metrics indicate the need for refactoring.

[SCREENSHOT S24]



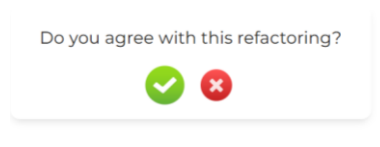
8.3 Accepting or Rejecting a Prediction

C) "Do you agree with this refactoring?" (right):

Two buttons are displayed:

Button	Icon	Action
Accept	Green checkmark ✓	Confirms that the prediction is correct
Reject	Red X ✗	Rejects the prediction — opens the Expert's Opinion section

[SCREENSHOT S25]



Step 5: If you agree with the prediction, click Accept (✓).

→ The prediction is confirmed. You can select another file.

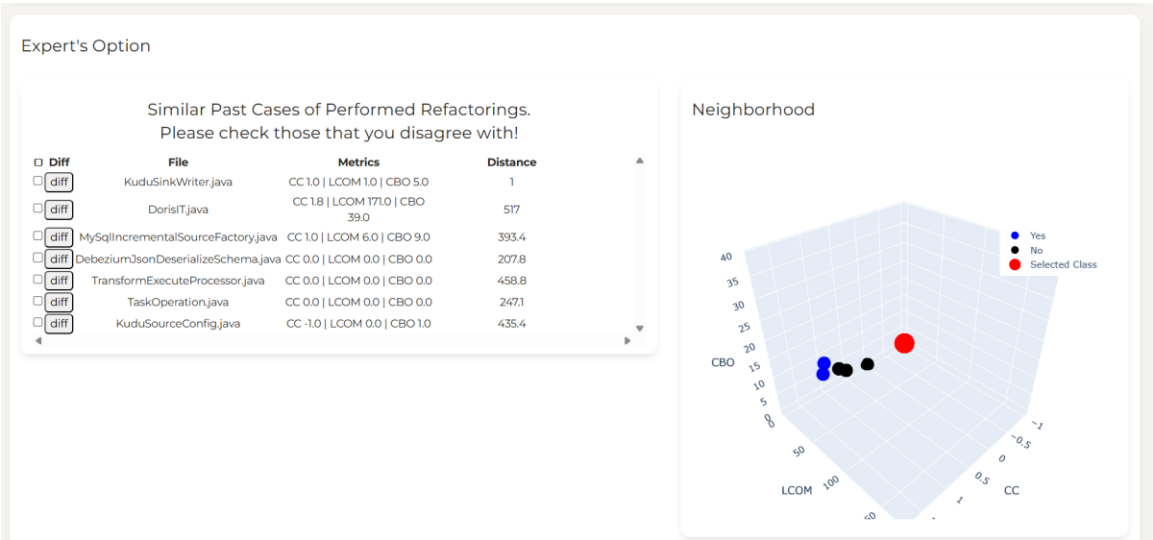
Step 6: If you disagree, click Reject (✗).

→ The "Expert's Opinion" section appears for providing feedback (see section 8.4).

8.4 Expert's Opinion

After clicking Reject, the "Expert's Opinion" section appears with the following:

[SCREENSHOT S26]



Similar Past Cases Table (Left)

Title: "Similar Past Cases of Performed Refactorings. Please check those that you disagree with!"

The table contains:

Column	Description
☐ (Checkbox)	Check the cases you disagree with
Diff	Button/link to view the commit diff
File	File name
Metrics	File metrics
Distance	Metric distance from the selected file (lower = more similar)

[SCREENSHOT S27]

Expert's Option

Similar Past Cases of Performed Refactorings.
Please check those that you disagree with!

☐ Diff	File	Metrics	Distance
☐ diff	KuduSinkWriter.java	CC 1.0 LCOM 1.0 CBO 5.0	1
☐ diff	DorisIT.java	CC 1.8 LCOM 171.0 CBO 39.0	517
☐ diff	MySQLIncrementalSourceFactory.java	CC 1.0 LCOM 6.0 CBO 9.0	393.4
☐ diff	DebeziumJsonDeserializeSchema.java	CC 0.0 LCOM 0.0 CBO 0.0	207.8
☐ diff	TransformExecuteProcessor.java	CC 0.0 LCOM 0.0 CBO 0.0	458.8
☐ diff	TaskOperation.java	CC 0.0 LCOM 0.0 CBO 0.0	247.1
☐ diff	KuduSourceConfig.java	CC -1.0 LCOM 0.0 CBO 1.0	435.4

Step 7: In the table, check the boxes for the cases you disagree with.

Step 8: You can click "Diff" on a row to see what changed in that commit (the Commit Diff viewer opens).

→ The code changes (diff) for that file are displayed.

[SCREENSHOT S28]

Diff

LOAD DIFF

Files changed (1)

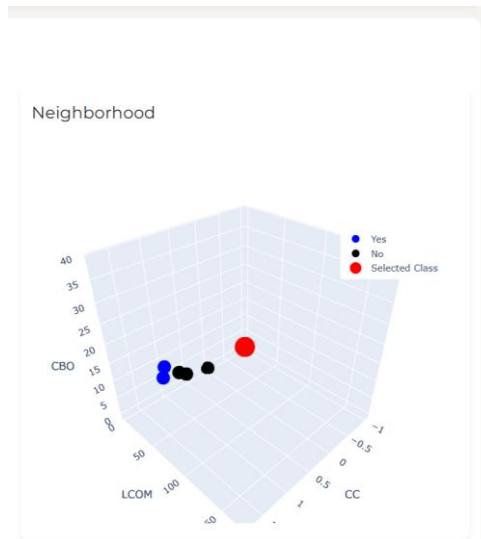
seatunnel-connectors-v2/connector-kudu/src/main/java/org/apache/seatunnel/connectors/seatunnel/kudu/sink/KuduSinkWriter.java CHANGED

```
@@ -17,34 +17,30 @@
17 17
18 18     package org.apache.seatunnel.connectors.seatunnel.kudu.sink;
19 19
20      - import org.apache.seatunnel.shade.com.typesafe.config.Config;
+ 20  + import org.apache.seatunnel.api.sink.SinkWriter;
21 21  -
22 21      import org.apache.seatunnel.api.table.type.SeaTunnelRow;
23 22      import org.apache.seatunnel.api.table.type.SeaTunnelRowType;
24 24  - import org.apache.seatunnel.connectors.seatunnel.common.sink.AbstractSinkWriter;
25 23      import org.apache.seatunnel.connectors.seatunnel.kudu.config.KuduSinkConfig;
26 24      import org.apache.seatunnel.connectors.seatunnel.kudu.kuduclient.KuduOutputFormat;
+ 25  + import org.apache.seatunnel.connectors.seatunnel.kudu.state.KuduCommitInfo;
+ 26  + import org.apache.seatunnel.connectors.seatunnel.kudu.state.KuduSinkState;
27 27
28 28      import lombok.NonNull;
29 29      import lombok.extern.slf4j.Slf4j;
30 30
31 31      import java.io.IOException;
+ 32  + import java.util.Optional;
```

Neighborhood Plot (Right)

Shows a neighborhood plot — how the selected file relates to similar files in a metric space. This helps understand why the model made its prediction.

[SCREENSHOT S29]



Saving Your Opinion

Step 9: After checking the cases you disagree with, click the save icon (floppy disk) in the "Save expert opinion" card.

→ Your opinion is saved. It can be used to retrain the model.

[SCREENSHOT S30]

Save expert opinion



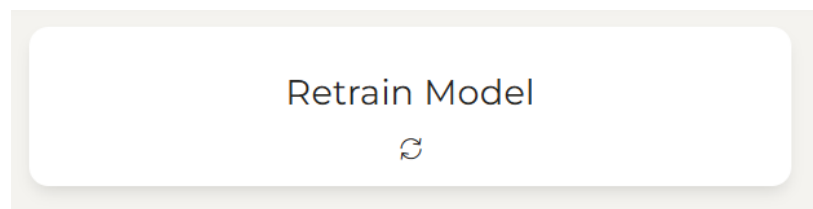
8.5 Retraining the Model

At the bottom of the page, the "Retrain Model" card allows you to retrain the ML model with updated data.

Step 10: Click the ↺ (refresh) icon in the "Retrain Model" card.

→ The model is retrained. This process may take several minutes. After completion, predictions are updated.

[SCREENSHOT S31]



NOTE: Retraining incorporates the expert opinions that were saved, improving future predictions.

9. Chart Interactions

All bar/line charts in the application support the same interactions:

Action	How	Result
Zoom In	Mouse wheel up	Focus on a narrower time range
Zoom Out	Mouse wheel down	Show a wider time range
Pan	Click + drag	Move the visible range
Reset Zoom	"Reset Zoom" button	Return to original view
Toggle Series	Click on legend	Show/hide a data series

Treemap Heatmaps (High TD Risk, Refactoring Urgency) support:

- Hover: Shows tooltip with information (name, probability/risk)
- Click: Selects a file for detailed analysis

10. Metrics Glossary

Detailed table of the metrics used by the application, what they measure, and what high values indicate:

Metric	Full Name	What It Measures	High Value Means
Complexity	Cyclomatic Complexity	Independent execution paths	Hard to understand/test code
DIT	Depth of Inheritance Tree	Inheritance depth	Complex class hierarchy
NOCC	Number of Child Classes	Direct subclasses	Many classes depend on it
RFC	Response for a Class	Methods that can be invoked	Complex interaction
LCOM	Lack of Cohesion of Methods	How unrelated methods are	Class does too many things
WMC	Weighted Methods per Class	Sum of method complexities	Too many complex methods
NOM	Number of Methods	Number of methods	Large class
MPC	Message Passing Coupling	Calls to other classes	High coupling
DAC	Data Abstraction Coupling	Use of other classes as types	Depends on many classes
CBO	Coupling Between Objects	Coupled classes	Changes may cascade widely
SIZE1	Size Metric 1	Lines of code	Large file
SIZE2	Size Metric 2	Methods and attributes	Complex structure

Key Terms:

Technical Debt: The implied cost of future rework caused by choosing suboptimal solutions

Interest Rate: The rate at which technical debt grows if left unaddressed

Cumulative Debt: The total debt accumulated over the project's lifetime

Refactoring: Restructuring existing code without changing its behavior to improve quality

Heatmap: A visualization where color intensity represents data values

Treemap: A visualization displaying hierarchical data as proportionally sized rectangles

High-TD: High Technical Debt — files with a high probability of significant debt

K Metric: A composite metric used to calculate debt per file

Appendix: Screenshot Checklist

The following table contains the full list of screenshots to be inserted into this manual, along with detailed instructions on what should be visible in each one.

ID	Page	Section	What Should Be Visible
S01	Login	2	Full Login page with Email, Password fields, Log In / Skip buttons
S02	Login	2	Login after failure — red notification "Invalid email or password"
S03	Dashboard	3	Full Dashboard page (full-page screenshot)
S04	Dashboard	3.1	Close-up: 3 summary cards (Projects, LoC, Refactorings)
S05	Dashboard	3.2	Close-up: Cumulative Debt chart with Reset Zoom button
S06	Dashboard	3.3	Close-up: 1 project row/card (metrics + 4 buttons)
S07	Dashboard	3.3	Close-up: The 4 action buttons (More, High TD, Urgent, Analyze)
S08	Dashboard	3.4	2 ranking cards: Cleaner Contributors + Most Refactoring
S09	Dashboard	4	"Add New Project" modal (URL, Refactoring Label, Add, Cancel)
S10	Dashboard	5	Yellow analysis notification "Debt analysis is running... XX%"
S11	More	6	Full Project More page
S12	More	6.1	Close-up: 5 cards (Cumul. Debt, Weeks Debt, IR, LoC, Refactorings)

S13	More	6.2	Close-up: Debt/IR chart with legend + Reset Zoom
S14	More	6.3	Metrics table (File, Debt, K, Complexity, RFC, MPC, NOM, etc.)
S15	High Risk	7	Full High TD Risk page
S16	High Risk	7.1	Close-up: Heatmap treemap (white→red rectangles)
S17	High Risk	7.1	Close-up: "Show all files" toggle (on/off)
S18	High Risk	7.2	Interpretation after clicking a file (path + chart)
S19	High Risk	7.3	"Root Causes" card (metric bar chart)
S20	High Risk	7.3	"Metrics Threshold" card (threshold table)
S21	Urgent	8	Full Refactoring Urgency page
S22	Urgent	8.1	Close-up: Refactoring probability heatmap (+ tooltip if possible)
S23	Urgent	8.2	Prediction metric cards (Recommended / No Need / %)
S24	Urgent	8.2	"Reason to Refactor" card (image)
S25	Urgent	8.3	"Do you agree?" card with Accept ✓ / Reject X
S26	Urgent	8.4	Full Expert's Opinion section
S27	Urgent	8.4	"Similar Past Cases" table (checkboxes, diff, file, metrics)
S28	Urgent	8.4	Commit Diff viewer (green/red diff)
S29	Urgent	8.4	"Neighborhood" card (scatter/network plot)
S30	Urgent	8.4	"Save expert opinion" card (floppy

			disk icon)
S31	Urgent	8.5	"Retrain Model" card (refresh icon)