Project Name	RTDIP Data Quality Checker
Online team meeting	https://fau.zoom-x.de/j/65297375649
Production system (if any)	...
Test system (if any)	...
GitHub repository	https://github.com/amosproj/amos2024ws01-rtdip-data-quality-checker
GitHub feature board	https://github.com/orgs/amosproj/projects/73/views/2
GitHub imp-squared backlog	https://github.com/orgs/amosproj/projects/74/views/1
Team T-shirt (white)	https://www.shirtinator.de/t-shirts/gestalten/t-shirt-bedrucken#/load/share/88a2f8c7-961f-4c63-a1bf-9461971dfdc0
Team T-shirt (black)	NA
Additional materials	...
Team maling list	oss-amos-proj1@lists.fau.de

Last Name	First Name	GitHub User Name	Email Address
Hoffmann	Dominik	dh1542	dominik.a.hoffmann@fau.de | dominik151099@outlook.de (github)
Katziuk	Avi	AviKatziuk	avi.katziuk@fau.de
B.	Timm	Timm638	timm638@gmail.com (For GitHub only)
Munz	Christian	chris-1187	c.munz@campus.tu-berlin.de (GitHub: christian.munz@posteo.de)
Tran	Minh Khue	kristen149	minh.khue.tran@fau.de
Baumgärtner	Lucca	luccalb	lucca.baumgaertner@fau.de
Moll	Leon	mollle	leonmariusmoll@gmail.com
Trost	Felipe	felipetrost	felipe.trost@gmail.com
Sanal	Mert	sanalmert	mert.sanal@campus.tu-berlin.de

#	Meeting Day	Product Owners	Software Developer	Release Manager	Scrum Master	Comment
1	2024-10-16	Lucca Baumgärtner	Everyone else		Avi Katziuk	
2	2024-10-23	Mert Sanal	Everyone else		Avi Katziuk	
3	2024-10-30	Lucca Baumgärtner	Everyone else	Timm	Avi Katziuk	
4	2024-11-06	Mert Sanal	Everyone else	Dominik	Avi Katziuk	
5	2024-11-13	Lucca Baumgärtner	Everyone else	Leon	Avi Katziuk	
6	2024-11-20	Mert Sanal	Everyone else	Christian	Avi Katziuk	
7	2024-11-27	Lucca Baumgärtner	Everyone else	Minh Khue	Avi Katziuk	Mid-term due
8	2024-12-04	Mert Sanal	Everyone else	Felipe	Avi Katziuk	
9	2024-12-11	Lucca Baumgärtner	Everyone else	Timm	Avi Katziuk	
10	2023-12-18	Mert Sanal	Everyone else	Dominik	Avi Katziuk	
11	2024-01-08	Lucca Baumgärtner	Everyone else	Leon	Avi Katziuk	
12	2024-01-15	Mert Sanal	Everyone else	Christian	Avi Katziuk	
13	2024-01-22	Lucca Baumgärtner	Everyone else	Minh Khue	Avi Katziuk	
14	2024-01-29	Mert Sanal	Everyone else	Felipe	Avi Katziuk	Demo day!
15	2024-02-05	Lucca Baumärtner	Everyone else	Timm	Avi Katziuk	Retrospective
Product owners, software developers, and Scurm Master are set and ideally don't change over time; the critical part is the Release Manager role you need to define here						

Goals	Deliver high quality software components for RTDIP by having a successfull PR into the main project	
	Forefilling the requirments of our industry partner in a structured and non-stressful way, e.g. not pulling all-nighters	
	Have a great time and learn something in the process	
Meeting norms	Mandatory	
	Punctual and reliable schedule (meetings at the same time every week so we can schedule our personal life and stuff)	
	Inform the team on the previous day if you can't attend	
	Try to be on time, don't wait for late joiners unless their input is critical	
Working norms	Try to find uniform decisions by discussing and prioritizing the IPs whishes	
	Don't expect last minute all nighters from your team members	
	Always get at least one review by another SD for your PR	
	Review (merge or postpone) open PRs by Tuesday 12am to give the RM enough time	
	Comply with code standards that we decide on as a team	
	Would be good to plan ahead when everyone can put the work in so we can coordinate and communicate in a productive way	
	Not commiting non compiling code	
	Use feature branches	
	Scheduling their working times is up to the individual	
Coordination norms		
	Developing a good and working release pipeline. From requirment to merge in master	
	Team meetings are led by the POs	
	Equal distribution of story points, considering last week's differences	
	Tasks can be picked freely by team members, if a task isn't assigned the POs can decide	
	If one has technical problems/bugs during their tasks, other developers should support via online platforms, TeamViewer or conduct peer review	
Communication norms	Slack for messaging, Zoom for Meetings/Pair Porgramming	
	Illness: Depending on the privacy preference of the person either slack channel or SM	
	Respond to direct mentions within one workday, have an emergency thread in slack	
	Have a FAQ in the documentation that is frequently updated	
Consideration norms		
	Devs, Scrum Master and POs should be equal in the hierachry. If someone has a concern one should address it	
Cont. improvement norms	Tracking progress in github project boards via achieved story points	
Rewards	Praise team members in Slack if you think they did a great job on something	
Sanctions	Create a Meme for the group and post it to Slack or someplace where we can collect them?	
Signatures		
Scrum Master	Avi Katziuk	
Product owner	Lucca Baumgärtner	
Product owner	Mert Sanal	
Software developer		
Software developer	Christian Munz	
Software developer	Domink Hoffmann	
Software developer	Felipe Trost	
Software developer	Leon Moll	
Software developer	Minh K. Tran	
	Continuous Improvement Nor	lo everyone, the link to join the Zoom meeting can be found

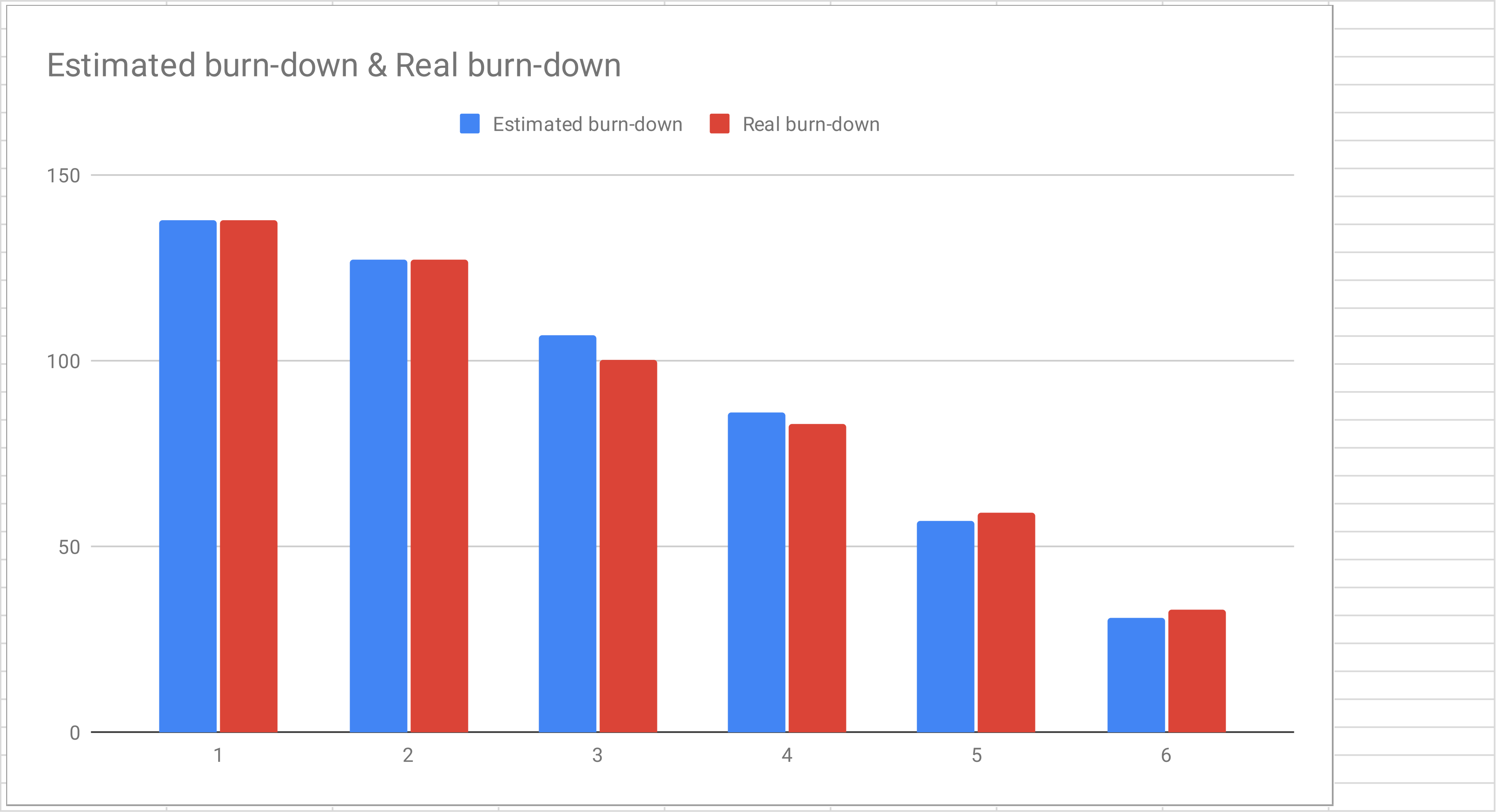
e	Project Mission
The Real-Time Data Ingestion Platform (RTDIP) by Shell is an open-source solution aimed at efficiently gathering and processing large-scale time-series data, such as information from millions of industrial sensors. It emphasizes scalability, innovation, and collaboration, with potential applications across various industries to enhance operational insights and decision-making.	To support the advancement of the Real-Time Data Ingestion Platform (RTDIP) by contributing to the development of innovative, open-source components focused on ensuring data quality. The mission includes creating tools to detect missing data, outliers, duplicates, and irregularities in real-time data streams, while aligning with RTDIP’s development guidelines to promote robust, scalable, and collaborative solutions.

Term	Definition
RTDIP	Real Time Data Ingestion Platform
A - D	
Data Sharing Mechanisms	Enables data sharing through approved architecture patterns, supporting streaming data or query-based batch transfers.
Delta Ingestion Engine	A cloud-based engine for processing time-series data from streaming endpoints (e.g., EventHub, Kafka) and files, ingesting it into a Delta Lakehouse.
Delta Lakehouse	A storage architecture that combines the reliability and scalability of data lakes with the performance and structure of data warehouses.
Destinations	Components that connect to sink/destination systems and write data to them.
E - G	
Edge	Components that will perform edge functionality such as connectors to protocols like OPC
Generative AI SQL Agent	An AI-powered tool that converts natural language into SQL queries for interacting with Databricks data.
I - M	
Ingestion Pipeline Framework	A modular framework that supports the creation, testing, and deployment of both streaming and batch data ingestion pipelines, allowing for customizable data processing workflows
Integration with Applications	Supports integration with tools like Digital Twin, C3.ai, SeeQ, and custom business solutions
LF Energy Integration	RTDIP is part of the LF Energy ecosystem, enabling energy and time-series data management.
Metadata Integration	Enables ingestion of metadata from sources such as PI, OPC UA, and APIs for better data context.
Monitoring	Components that are designed to analyze and assess the quality of datasets by detecting and identifying issues, particularly related to missing or anomalous data. These components do not modify or alter the data but provide insights for users to address data quality issues.
Manipulating	Components that are designed to transform, preprocess, and optimize datasets for analysis and machine learning tasks. These components modify the data to ensure it is clean, structured, and ready for further processing.
P - S	
Python SDK	A software development kit for programmatic data access, retrieval, and analysis in the Delta Lakehouse.
REST APIs	APIs for interacting with the Delta Lakehouse via HTTP requests.
Scalability	Designed to scale for large-scale ingestion and processing across numerous sensors and data sources.
Security Controls	Implements robust measures to ensure data protection and compliance.
Shell Contribution	Foundational time-series ingestion capability contributed by Shell, managing data from over three million sensors.
Streaming Endpoints	Real-time data sources such as EventHub and Kafka for data ingestion.
Sources	Connectors to source systems
T - Z	
Transformers	Components perform transformations on data, including data cleansing, data enrichment, data aggregation, data masking, data encryption, data decryption, data validation, data conversion, one hot encoding
Utilities	Components that perform utility functions such as logging, error handling, data object creation, authentication, maintenance

Sprint #	Sprint goal
1	None
2	None
3	None
4	Optional
5	Build a product demo for the mid-project & final release
6	Finalize demo and various components
7	Improve testing and apply Shells feedback
8	Continue refactoring and improve test quality
9	Standardizing tests & adding functionality
10	Coming up with new tasks based on workshop outcome
11	Brainstorming for the Demo and finishing up Documentation
12	Adding new components to ensure accurate workflow for SDs & initiliazing preparations for the demo
13	Integrate latest IP feedback and prepare demo poster/video
14	
15	

Sprint	Goal	Feature Name	Est. Size	Est. Remaining	Real Size	Real Remaining
Release						
Total			138	31		
Sprints						
1	**Issues Finished in Sprint No. #1**		11	138	11	138
2	**Issues Finished in Sprint No. #2**		20	127	27	127
3	**Issues Finished in Sprint No. #3**		21	107	17	100
4	**Issues Finished in Sprint No. #4**		29	86	24	83
5	**Issues Finished in Sprint No. #5**		26	57	26	59
6	**Issues Finished in Sprint No. #6**		31	31	31	33
Features						
1	**Issues Finished in Sprint No. #1**					
		Duplicate Detection	8		8	
		Fix Broken Virtual Environment	3		3	
2	**Issues Finished in Sprint No. #2**					
		Create Software Bill of Materials	1		1	
		Create Software Architecture Diagram	3		5	
		Anomaly Detection	3		8	
		Explore the Test Data and Brainstorm RTDIP Component Ideas	5		5	
		Identify Missing Data	8		8	
3	**Issues Finished in Sprint No. #3**					
		Create a Test Pipeline to Run During Release	5		1	
		Clean Data Based on Interval/Pattern	8		8	
		Normalization of Data	8		8	
4	**Issues Finished in Sprint No. #4**					
		Time Series Prediction Using ARIMA	13		8	
		Clean Data Based on Interval/Pattern	8		8	

Sprint	Goal	Feature Name	Est. Size	Est. Remaining	Real Size	Real Remaining
		Normalization of Data	8		8	
5	**Issues Finished in Sprint No. #5**					
		Time Series Prediction with Linear Regression	8		8	
		Missing Value Imputation	13		13	
		Validation of Value Ranges	3		3	
		Flatline Detection	2		2	
6	**Issues Finished in Sprint No. #6**					
		Reduce Number of Parameters Needed to Use ArimaPrediction Effectively	8		8	
		Interval Filtering not Working for EventTime Column of Type 'datetime'	2		2	
		One-Hot Encoding	3		3	
		Homework - User/Design/Build Documentation	5		5	
		Prepare RTDIP Demo	8		8	
		Data Binning	5		5	

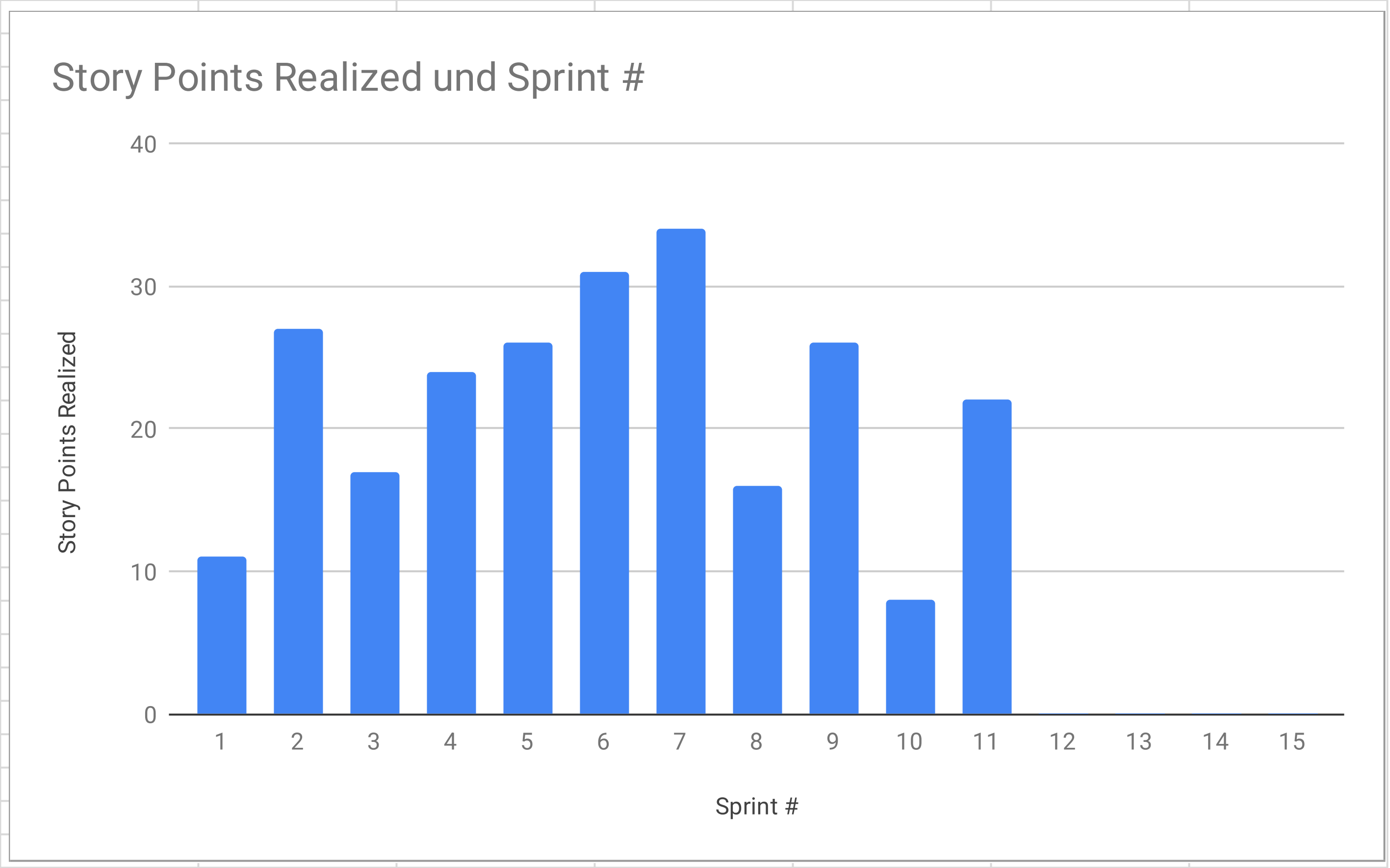

Estimated burn-down & Real burn-down
🟦 Estimated burn-down
🟥 Real burn-down
150
100
50
0
1
2
3
4
5
6

Sprint	Goal	Feature Name	Est. Size	Est. Remaining	Real Size	Real Remaining
Release						
Total			99	99		
Sprints						
7	**Planned Issues for Sprint No. #7**		31	99	34	99
8	**Planned Issues for Sprint No. #8**		16	68	16	65
9	**Planned Issues for Sprint No. #9**		26	52	26	49
10	**Planned Issues for Sprint No. #10**		5	26	8	23
11	**Planned Issues for Sprint No. #11**		21	21	22	15
12	**Planned Issues for Sprint No. #12**		0	0	0	-7
13	**Planned Issues for Sprint No. #13**		0	0	0	-7
14	**Planned Issues for Sprint No. #14**		0	0	0	-7
15	**Planned Issues for Sprint No. #15**		0	0	0	-7
Features						
7	**Planned Issues for Sprint No. #7**					
		Store Monitoring Outputs in a Standardized Format	13		13	
		Apply Feedback for Duplicate Detection	2		1	
		Apply Feedback for Interval Filtering	1		1	
		Apply Feedback for Value Range Check	1		2	
		Apply Feedback for Missing Data Identification	1		1	
		Apply Feedback on Project Structure	2		5	
		Unified Input Data Validation	8		8	
		Advanced Duplicate Detection	2		2	
		Apply Feedback for Anomaly Detection	1		1	
8	**Planned Issues for Sprint No. #8**					
		Fix broken API test	5		5	
		Value Range Validation: Refactor Unit Tests	3		3	
		Flatline detection: Refactor unit tests	3		3	
		Missing Data Detection: Refactor Unit Tests	5		5	

Sprint	Goal	Feature Name	Est. Size	Est. Remaining	Real Size	Real Remaining
9	**Planned Issues for Sprint No. #9**					
		Dimensionality Reduction	5		5	
		Duplicate detection: Refactor unit tests	3		3	
		Linear regression: Refactor unit tests	3		3	
		ARIMA: Refactor unit tests	5		5	
		Anomaly detection: Refactor unit tests	3		2	
		Interval filtering: Refactor unit tests	3		2	
		Missing Value Imputation: Refactor unit tests	3		5	
		Restore Deliverables Folder	1		1	
10	**Planned Issues for Sprint No. #10**					
		Finish Integrating ARIMA Functionality of statsmodels into RTDIP	5		8	
11	**Planned Issues for Sprint No. #11**					
		Finish implementation of feedback and our first major release (PR #57)	3		3	
		De/normalization: refactor unit tests	3		5	
		Put Value Range Check Component into Action	3		3	
		Remove flatlining datapoints	3		3	
		Refine Product Glossary	3		2	
		Finalize Documentation	3		3	
		Deciding Which Use-Case to Present for Demo-Day	3		3	
12	**Planned Issues for Sprint No. #12**					
		Demo pipeline of multiple components	8			
		New Component: Moving Average	5			
		New Component: Gaussian Smoothing	5			
		Create a new (Final)PR for Shell to Review	2			
		Write an article about RTDIP & AMOS	5			
		Link all documentation in Planning Document	3			
		New Component: KNN	8			

Sprint	Goal	Feature Name	Est. Size	Est. Remaining	Real Size	Real Remaining

Sprint #	Story Points Realized
1	11
2	27
3	17
4	24
5	26
6	31
7	34
8	16
9	26
10	8
11	22
12	0
13	0
14	0
15	0
	PLEASE CREATE THE VELOCITY CHART ON A NEW TAB USING THE DATA FROM THIS TAB


Story Points Realized und Sprint #
40
30
20
10
0
Story Points Realized
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Sprint #

#	Feature Definition of Done	Sprint Release Definition of Done	Project Release Definition of Done
1	Test cases have been created and are running successfully	Change log has been updated	Approval from IP
2	Documentation for the new component was added	Test suite running successfully	Successful merge into original RTDIP repository
3	Github Actions are running without errors	Bill of materials up-to-date	

Type	Link / reference
	The components that we developed can be found under SDK-> Pipelines -> Components -> Data Quality
Github Pages	https://amosproj.github.io/amos2024ws01-rtdip-data-quality-checker/

#	Context	Name	Version Range	License	Comment
1	conda-forge	databricks-sql-connector	>=3.1.0,<4.0.0	Apache 2.0	SQL connector for Databricks
2	conda-forge	azure-identity	>=1.12.0,<2.0.0	MIT	Identity management for Azure
3	pip	pandas	>=1.5.2,<2.2.0	BSD 3-Clause	Data manipulation library
4	conda-forge	jinja2	>=3.1.4,<4.0.0	BSD 3-Clause	Template engine for Python
5	conda-forge	importlib_metadata	>=7.0.0	MIT	Metadata for Python packages
6	conda-forge	semver	>=3.0.0,<4.0.0	MIT	Semantic versioning library
7	conda-forge	xlrd	>=2.0.1,<3.0.0	MIT	Library for reading Excel files
8	conda-forge	grpcio	>=1.48.1	Apache 2.0	gRPC library for Python
9	conda-forge	grpcio-status	>=1.48.1	Apache 2.0	gRPC status library
10	conda-forge	googleapis-common-protos	>=1.56.4	Apache 2.0	Common protobufs for Google APIs
11	pip	langchain	>=0.2.0,<0.3.0	MIT	Framework for LLMs
12	pip	langchain-community	>=0.2.0,<0.3.0	MIT	Community contributions to LangChain
13	conda-forge	openai	>=1.13.3,<2.0.0	MIT	OpenAI API client
14	conda-forge	pydantic	>=2.6.0,<3.0.0	MIT	Data validation library
15	conda-forge	pyspark	>=3.3.0,<3.6.0	Apache 2.0	Spark library for Python
16	conda-forge	delta-spark	>=2.2.0,<3.3.0	Apache 2.0	Delta Lake integration with Spark
17	pip	dependency-injector	>=4.41.0,<5.0.0	MIT	Dependency injection framework
18	pip	databricks-sdk	>=0.20.0,<1.0.0	Apache 2.0	SDK for Databricks services
19	conda-forge	azure-storage-file-datalake	>=12.12.0,<13.0.0	MIT	Azure Data Lake Storage client
20	conda-forge	azure-mgmt-storage	>=21.0.0	MIT	Azure Storage management client
21	pip	azure-mgmt-eventgrid	>=10.2.0	MIT	Azure Event Grid management client
22	conda-forge	boto3	>=1.28.2,<2.0.0	Apache 2.0	AWS SDK for Python
23	pip	hvac	>=1.1.1	MPL 2.0	HashiCorp Vault client
24	conda-forge	azure-keyvault-secrets	>=4.7.0,<5.0.0	MIT	Azure Key Vault secrets management
25	pip	web3	>=6.18.0,<7.0.0	MIT	Ethereum blockchain library
26	conda-forge	polars[deltalake]	>=0.18.8,<1.0.0	MIT	DataFrame library with Delta Lake support
27	conda-forge	delta-sharing	>=1.0.0,<1.1.0	Apache 2.0	Delta Sharing library
28	conda-forge	xarray	>=2023.1.0,<2023.8.0	BSD 3-Clause	N-dimensional array library
29	conda-forge	ecmwf-api-client	>=1.6.3,<2.0.0	Apache 2.0	ECMWF API client
30	conda-forge	netCDF4	>=1.6.4,<2.0.0	BSD 3-Clause	NetCDF file reading/writing
31	conda-forge	joblib	>=1.3.2,<2.0.0	BSD 3-Clause	Lightweight pipelining library
32	pip	sqlparams	>=5.1.0,<6.0.0	MIT	SQL query parameters library
33	pip	entsoe-py	>=0.5.10,<1.0.0	MIT	ENTSOE API client
34	conda-forge	pytest	==7.4.0	MIT	Testing framework
35	conda-forge	pytest-mock	==3.11.1	MIT	Mocking for pytest
36	conda-forge	pytest-cov	==4.1.0	MIT	Coverage reporting for pytest
37	conda-forge	pylint	==2.17.4	GPL 2.0	Static code analysis for Python
38	conda-forge	pip	>=23.1.2	MIT	Python package installer
39	conda-forge	turbodbc	==4.11.0	MIT	ODBC interface for Python
40	conda-forge	numpy	>=1.23.4,<2.0.0	BSD 3-Clause	Numerical computing library
41	conda-forge	oauthlib	>=3.2.2,<4.0.0	MIT	OAuth library
42	conda-forge	cryptography	>=38.0.3	MIT	Cryptography library

#	Context	Name	Version Range	License	Comment
43	conda-forge	fastapi	>=0.110.0,<1.0.0	MIT	Fast web framework
44	conda-forge	httpx	>=0.24.1,<1.0.0	MIT	HTTP client for Python
45	conda-forge	openjdk	>=11.0.15,<12.0.0	N/A	OpenJDK Java runtime
46	conda-forge	mkdocs-material	==9.5.20	MIT	Material theme for MkDocs
47	conda-forge	mkdocs-material-extensions	==1.3.1	MIT	Extensions for MkDocs
48	conda-forge	mkdocstrings	==0.25.0	MIT	Documentation generation
49	conda-forge	mkdocstrings-python	==1.10.8	MIT	Python support for mkdocstrings
50	conda-forge	mkdocs-macros-plugin	==1.0.1	MIT	Macros for MkDocs
51	conda-forge	mkdocs-autorefs	>=1.0.0,<1.1.0	MIT	Automatic references for MkDocs
52	conda-forge	pygments	==2.16.1	BSD 2-Clause	Syntax highlighting library
53	conda-forge	pymdown-extensions	==10.8.1	MIT	Extensions for Markdown
54	conda-forge	pygithub	>=1.59.0	MIT	GitHub API client
55	conda-forge	pyjwt	>=2.8.0,<3.0.0	MIT	JSON Web
56	conda-forge	conda	>=24.9.2	BSD 3-Clause	Package installer
57	pip	statsmodels	>=0.14.1,<0.15.0	BSD 3-Clause	Statistical Models for Data Forecasting
58	pip	pmdarima	>=2.0.4	MIT	Used as Wrapper for statsmodels

Last Name	**First Name**	**Value**					
				7.40	**NOK**		
Katziuk	**Avi**						
B.	**Timm**	8					
Munz	**Christian**	5		0	No size		
Tran	**Minh Khue**	8		1	Trivial size		
Baumgärtner	**Lucca**			2	Small size		
Moll	**Leon**	8		3	Medium size		
Trost	**Felipe**	8		5	Large size		
Sanal	**Mert**			8	Very large size		
Hoffmann	**Dominik**	5		13	Too large (size)		
How to play planning poker							
1. Everyone type their number into their value field, don't hit return yet							
2. Someone, perhaps a product owner, count down 3.. 2.. 1..							
3. Then, everyone hit return to submit their value							