Project Name	RTDIP Timeseries Forecasting
Online team meeting	https://fau.zoom-x.de/j/65502405196?pwd=8H73lyixQfqKrnO8Eb47kawnuKCnGp.1
Production system (if any)	
Test system (if any)	
GitHub repository	https://github.com/amosproj/amos2025ws03-rtdip-timeseries-forecasting
GitHub feature board	https://github.com/orgs/amosproj/projects/91/
GitHub imp-squared backlog	https://github.com/orgs/amosproj/projects/96
Team T-shirt (white)	https://www.shirtinator.de/s/QWcDXTKGS72ISnrRNZq1hg
Team T-shirt (black)	https://www.shirtinator.de/s/0yx3duRjSzW-5hP32KmZ1w
Additional materials	https://discord.gg/KJXGmjcs
Team maling list	oss-amos-projX@lists.fau.de
Happines Index App	https://happy-amos.appspot.com/Courses
Planning Poker	https://planningpokeronline.com/DMhnF5cOAk9ffF5jsRf2/

Last Name	First Name	GitHub User Name	Email Address
Böhm	Luca	qw3rat	luca.boehm@fau.de
Meusling	Patrick	SirGankalot	meusling@campus.tu-berlin.de
Arifin	Hafidz	zenzeii	h.arifin@campus.tu-berlin.de
Huy	Christoph	Perimora	christoph.huy@campus.tu-berlin.de
Selbig	Simon	simonselbig	simon.selbig@gmx.de
Haseeb	Abdul	abdulhaseeb-se	abdul.ah.haseeb@fau.de
Pohnke	Hannes	hpfref	hannes.j.pohnke@gmail.com
Jokiel	Rene	BelmontR	rene.jokiel@fau.de
Khabouze	Mehdi	Mehdi-kbz	khabouze@campus.tu-berlin.de

		Product Owner					
#	Meeting Day	Review	Planning	Software Developer	Release Manager	Scrum Master	Comment
1	2025-10-15	Patrick Meusling	Hafidz Arifin	Everyone else	N/A	Luca	
2	2025-10-22	Hafidz Arifin	Patrick Meusling	Everyone else	Christoph	Luca	
3	2025-10-29	Patrick Meusling	Hafidz Arifin	Everyone else	Mehdi Khabouze	Luca	
4	2025-11-05	Hafidz Arifin	Patrick Meusling	Everyone else	Rene Jokiel	Luca	
5	2025-11-12	Patrick Meusling	Hafidz Arifin	Everyone else	Abdul Haseeb	Luca	
6	2025-11-19	Hafidz Arifin	Patrick Meusling	Everyone else	Hannes Pohnke	Luca	
7	2025-11-26	Patrick Meusling	Hafidz Arifin	Everyone else	Simon Selbig	Luca	Mid-term due
8	2025-12-03	Hafidz Arifin	Patrick Meusling	Everyone else	Rene Jokiel	Luca	
9	2025-12-10	Patrick Meusling	Hafidz Arifin	Everyone else	Mehdi Khabouze	Luca	
10	2026-01-11	Hafidz Arifin	Patrick Meusling	Everyone else	Abdul Haseeb	Luca	
11	2026-01-18	Patrick Meusling	Hafidz Arifin	Everyone else	Hannes Pohnke	Luca	
12	2026-01-25	Hafidz Arifin	Patrick Meusling	Everyone else	Simon Selbig	Luca	
13	2026-02-01	Patrick Meusling	Hafidz Arifin	Everyone else	Mehdi Khabouze	Luca	
14	2026-02-08	Hafidz Arifin	Patrick Meusling	Everyone else	Christoph	Luca	Demo day!
15	2026-02-15	Patrick Meusling	Hafidz Arifin	Everyone else	Hannes Pohnke	Luca	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							

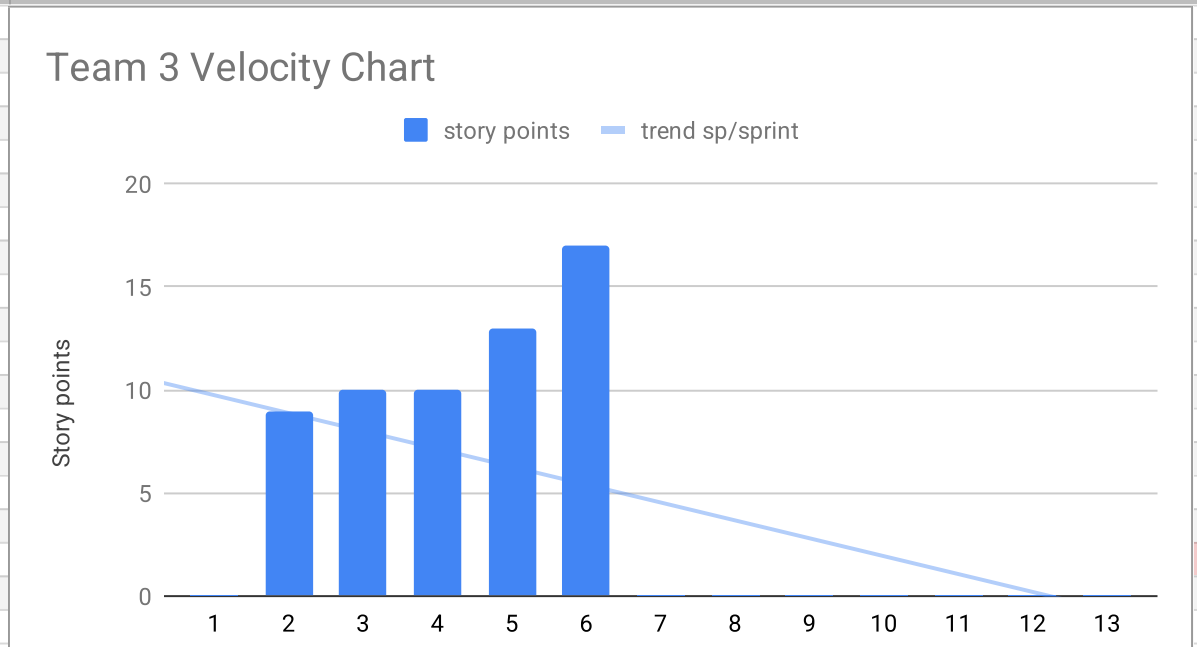
Goals	Aquire new skills
	Produce a functioning and valuable product (a 1.0)
Meeting norms	We respect the opinions of others and we show up on time
	Summary of Partner Meeting required
Working norms	Clean, testable code with clear commit messages, code comments
	We respect other people's work
Coordination norms	Seperate branches for issues, second person review
	We balance workload among the team
Communication norms	Daily check the corresponding channels (discord)
	We communicate constructively, make sure to communicate possible problems (absence, etc.)
Consideration norms	We discuss issues openly
	We vote in case we can't reach a consensus
Cont. improvement norms	We encourage critique and improvement efforts
	Encourage reaching out for help (use the strength/skills of others)
Rewards	We treat ourselfes to a sweet of choice for good work
Sanctions	Push Ups (amount decided on the case)
Signatures	
Scrum Master	Luca Böhm
Product owner	Patrick Meusling
Product owner	Hafidz Arifin
Software developer	Abdul Haseeb
Software developer	Rene Jokiel
Software developer	Simon Selbig
Software developer	Hannes Pohnke
Software developer	Christoph Huy
Software developer	Mehdi Khabouze

Product Vision	Project Mission
To empower organizations with open-source, scalable, and transparent forecasting capabilities that enable data-driven decision-making across industries. By extending RTDIP with advanced forecasting and anomaly detection components as well as sector specific datasets, we envision a future where time series insights such as trends, seasonality, and predictive patterns are seamlessly integrated into real-time data pipelines. This will allow businesses to understand historical data, anticipate future behavior, optimize operations, and improve efficiency using accessible, community-driven, and production-ready forecasting tools.	To design, develop, and contribute open-source forecasting components for the RTDIP platform that enable trend analysis, anomaly detection, and predictive modeling on time series data. Our mission within this project is to research and implement forecasting techniques using Python and Apache Spark, validate and enricht them with real-world datasets, and ensure they meet RTDIP's modular, tested, and well-documented standards. By doing so, we will enhance RTDIP's functionality and provide the open-source community and industry users such as Shell with reliable, scalable forecasting capabilities that integrate seamlessly into existing data pipelines.

Term	Definition
RTDIP (Real-Time Data Ingestion Platform)	An open-source project under the Linux Foundation designed to simplify the ingestion, transformation, and storage of data from various sources using scalable cloud technologies.
Forecasting	The process of predicting future values or trends based on historical time series data.
Time Series Data	Data collected over time at regular intervals, often used to analyze patterns such as trends and seasonality.
Apache Spark	An open-source distributed computing system that provides fast data processing for large-scale datasets.
PySpark	The Python API for Apache Spark, allowing developers to write Spark applications using Python.
Delta Lake	A storage layer that brings reliability, consistency, and performance to data lakes by supporting ACID transactions and schema enforcement.
Anomaly Detection	The identification of unusual patterns or outliers in data that do not conform to expected behavior.
Predictive Modeling	A statistical or machine learning approach that uses historical data to predict future outcomes.
Time Series Decomposition	The process of breaking down a time series into its core components such as trend, seasonality, and residuals.
ETL (Extract, Transform, Load)	A data pipeline process that extracts data from sources, transforms it into a suitable format, and loads it into a storage system.
Open Source	Software that is freely available to use, modify, and distribute, typically developed collaboratively by a community.
Linux Foundation	A nonprofit organization that supports open-source software projects and fosters collaboration across industries.
EasyCLA (Easy Contributor License Agreement)	A system used by the Linux Foundation to manage contributor license agreements, ensuring legal compliance for open-source contributions.
Modular Architecture	A software design principle that divides a system into separate, interchangeable components that can be developed and maintained independently.
Unit Testing	The practice of testing individual units or components of software to ensure they work as intended.
Documentation	Written descriptions and guides that explain the design, usage, and functionality of a software system.
Data Pipeline	A set of processes that move, transform, and store data from one system to another in a structured and automated way.
Scalability	The ability of a system to handle increasing amounts of work or data by adding resources.
Contribution Guidelines	A set of rules and best practices that contributors must follow to ensure consistency and quality in open-source projects.

Sprint #	Sprint goal
1	None
2	None
3	None
4	Optional
5	Establish a working foundation for the forecasting pipeline
6	Integrate the pipeline into the existing software
7	Advance the pipeline
8	Add more Datasets to the pipeline
9	Testing and Analysing the new Datasets
10	Adding new and diffrent models to the pipeline
11	Testing and Analysing the new Models
12	Smaller features for more/diffrent preprocessing etc.
13	Finishing the product so its ready
14	Merging the product with the partner, coordinating the merge (they said this might take a bit of time)
15	Backup Week

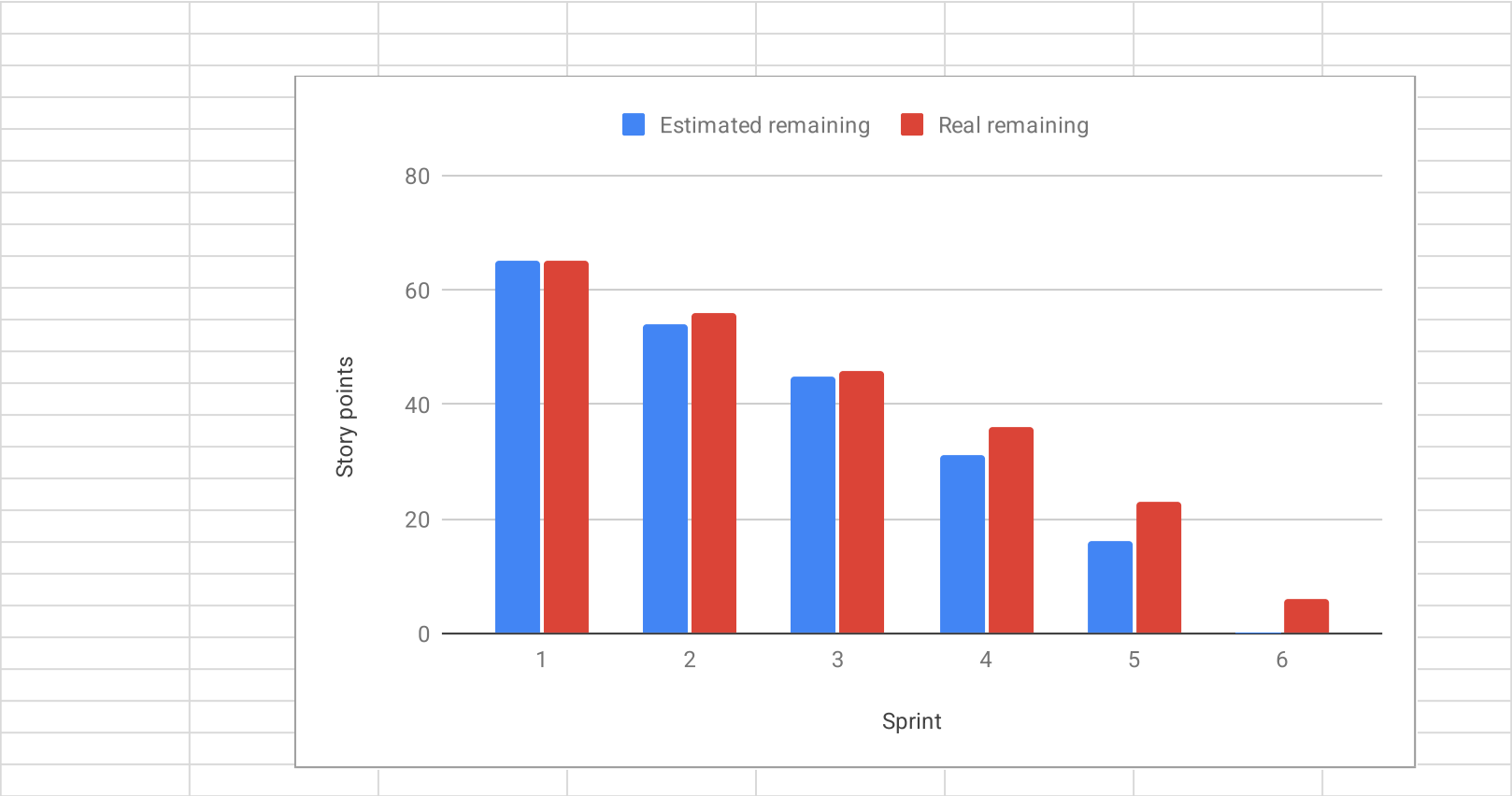
Sprint #	Story Points Realized
1	0
2	9
3	10
4	10
5	13
6	17
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
	PLEASE CREATE THE VELOCITY CHART ON A NEW TAB USING THE DATA FROM THIS TAB


Team 3 Velocity Chart
story points
trend sp/sprint
Story points
20
15
10
5
0
1
2
3
4
5
6
7
8
9
10
11
12
13

Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
Release						
Total			65	65		
Sprints						
1	None		0	65	0	65
2	None		11	54	9	56
3	None		9	45	10	46
4	Optional		14	31	10	36
5	Establish a working foundation for the forecasting pipeline		15	16	13	23
6	Integrate the pipeline into the existing software		16	0	17	6
Features						
1	**None**					
2	**None**					
		Run the software on your own machine	1		1	
		Understanding the rtdip software	2		2	
		Software Architecture	2		2	
		Find open source time series datasets	2		2	
		Initialize software bill of materials	2		2	
		Research about time series forecasting fundamentals	2		NotCompleted	
3	**None**					
		Research Possible Models Useful for Shell Dataset	2		2	
		Update our Forked Repository with Latest RTDIP Release	1		1	
		Prepare and Demo Project Build Process	1		2	
		Perform Data Exploration on Shell Dataset	3		3	
		Research about time series forecasting fundamentals	2		2	
4	**Optional**					
		Getting to Know HPC of AMOS	2		NotCompleted	
		Perform Data Exploration on Opensource Dataset 1	2		3	
		Preprocess Shell Dataset	3		3	
		Implementing a basic Model pipeline	2		2	
		Research Possible Models Userful fo Opensource Dataset	2		NotCompleted	
		CI/CD Pipeline fail bug	3		2	
5	**Establish a working foundation for the forecasting pipeline**					
		Choosing Models, Datasets, Error Measure Metrics	2		2	
		Create Build Process Video	2		2	
		Model Training and Evaluation with Shell Dataset	3		3	
		Visualization of Results	2		1	
		Preprocess Opensource Dataset 1	3		3	
		PO Homework	3		2	

Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
6	**Integrate the pipeline into the existing software**					
		Model Deployment Pipeline into RTDIP	3		3	
		User/Design/Build Documentation	2		2	
		Repository Structure Refactor	2		2	
		Update Software Architecture	2		2	
		Set Up Unified Development Environment	3		5	
		Collect and Document Customer Feedback on Project Progress	1		NotCompleted	
		Preprocess Opensource SCADA	3		3	
		PLEASE CREATE THE BURNDOWN CHART ON A NEW TAB USING THE DATA FROM THIS TAB				

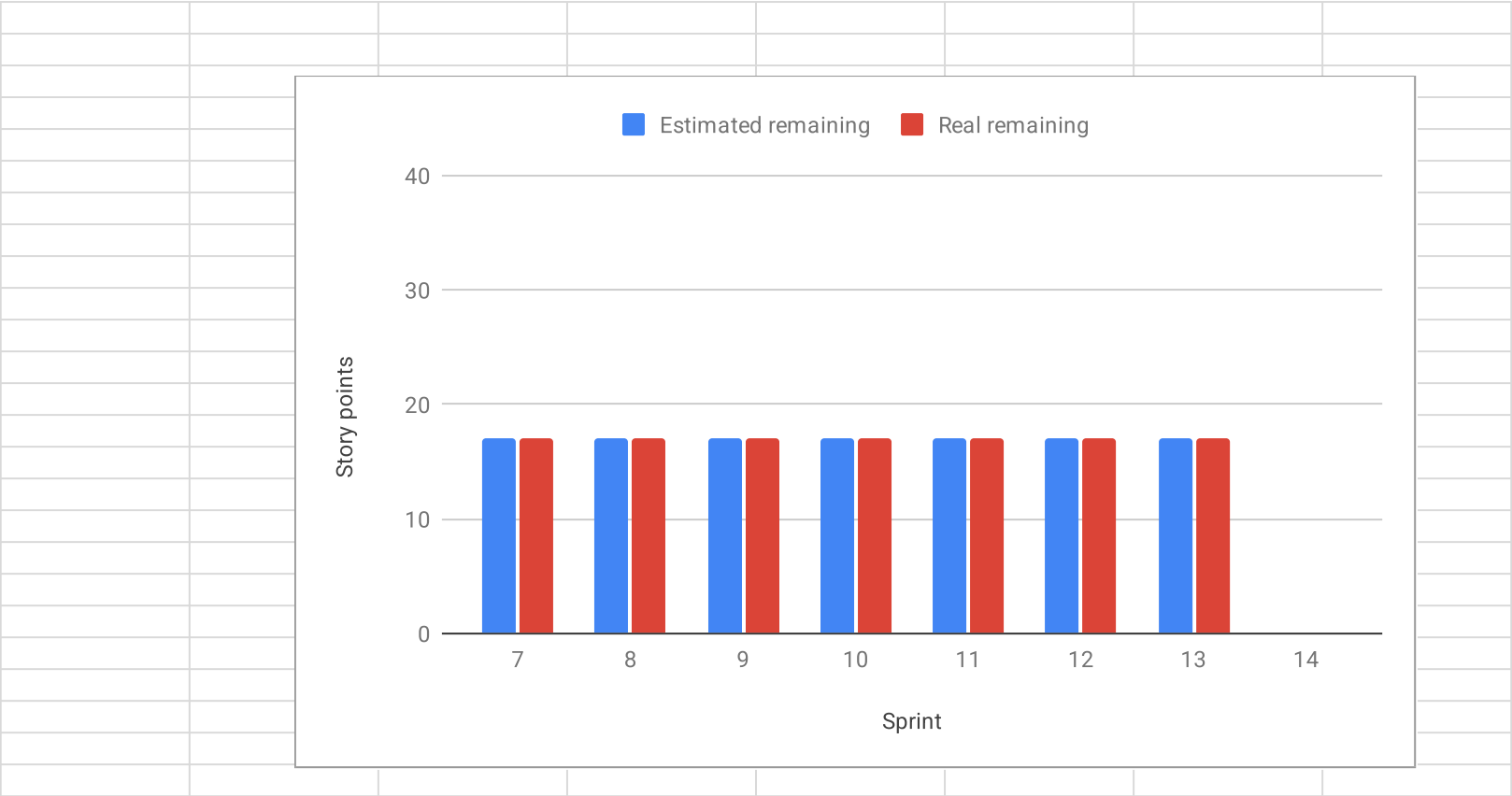
Mid Burndown Chart



Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
Release						
Total			17	17	0	0
Sprints						
7	Advance the pipeline		17	17	0	17
8	Add more Datasets to the pipeline		0	17	0	17
9	Testing and Analysing the new Datasets		0	17	0	17
10	Adding new and diffrent models to the pipeline		0	17	0	17
11	Testing and Analysing the new Models		0	17	0	17
12	Smaller features for more/diffrent preprocessing etc.		0	17	0	17
13	Finishing the product so its ready		0	17	0	17
14	Merging the product with the partner, coordinating the merge (they said this might take a bit of time)		0	17	0	17
Features						
7	**Advance the pipeline**					
		Collect and Document Customer Feedback on Project Progress	1			
		Adjust and Fix CI/CD Pipeline	3			
		Model Optimization and Improvement	3			
		Implementation of Time Series Decomposition	3			
		Implementation of Anomaly Detection	5			
		Model Training and Evaluation of Dataset SCADA	2			
8	**Add more Datasets to the pipeline**					

Sprint	Goal	Feature Name	Est. size	Est. remaining	Real size	Real remaining
		PLEASE CREATE THE BURNDOWN CHART ON A NEW TAB USING THE DATA FROM THIS TAB				

Final Burndown Chart



#	Feature Definition of Done	Sprint Release Definition of Done	Project Release Definition of Done
	Acceptance criteria are met.		
	Work are pushed to the Github repository.		
	Create a branch for each backlog items (coding)		
	A pull request is created for each related branch.		
	The work products in the pull requests are reviewed.		
	The corresponding branches are merged and closed.		
	The bill of materials section of the planning documents is updated.		
	Work needs to be documented in the corresponding wiki section		
	For new features unit test have to be written.		
	Update our forked repository with the latest RTDIP release.		
	If the task involves coding, the implementation is integrated into the RTDIP framework and verified to function correctly within it.		
	All unit tests must pass successfully in the continuous integration (CI) pipeline.		
	All assignees must communicate and coordinate with each other to complete the task.		
		Release candidates with a working and meaningful update to the previous sprint is tagged.	
		Previously established features must continue to work.	
			The RTDIP forecasting components can be successfully built and deployed within the RTDIP environment.
			All automated tests and validation checks for the developed components pass without errors.
			The implemented forecasting features have been tested end-to-end and verified through a basic user workflow.
			Developer documentation is complete, clearly describing the architecture, setup, and contribution process for RTDIP.
			User documentation (usage examples, configuration steps, expected outputs) is finalized and up to date.
			The final release has been reviewed and approved by all project team members and the industry partner (Shell).

Type	Link / reference
User Documentation	https://github.com/amosproj/amos2025ws03-rtdip-timeseries-forecasting/wiki/Documentation#user-documentation
Design Documentation	https://github.com/amosproj/amos2025ws03-rtdip-timeseries-forecasting/wiki/Documentation#design-documentation
Build & Deploy Documentation	https://github.com/amosproj/amos2025ws03-rtdip-timeseries-forecasting/wiki/Documentation#build-and-deploy-documentation

Project	BYP - Forecasting and Anomaly Detection
Version	0.4.1 (Sprint 04)
Date	06.10.2025
Last updated	06.10.2025
Environment	Development
Authors	Christoph Haz

System Components

Name	Category	Source	Description
Forecasting	ML Component	src/sdk/python/tsdp_sdk/pipelines/forecasting	Train / Prediction with Forecasting Models.
Transformer	ML Pre-Processing	src/sdk/python/tsdp_sdk/pipelines/transformers/spark/machine_learning	Data / Feature pre-processing for ML components.
Data Manipulator	Data Quality	src/sdk/python/tsdp_sdk/pipelines/data_quality/data_manipulation	Alters data to ensure data quality.
Monitor	Data Quality	src/sdk/python/tsdp_sdk/pipelines/data_quality/monitoring	Monitors quality of time-series data.
Source	Data Extraction	src/sdk/python/tsdp_sdk/pipelines/sources	Ingest data source.
Data Storage	Data Storage	external (delta-spark)	Provides raw time-series data for ingestion and processing.
Upcoming:			
Anomaly Detection	ML Component	TBA	Detects anomalies in time-series predictions.

ML Artifacts

Name	Category	Source	Description
TBA			

First Level Dependencies and Runtimes

Name	Category	Version	License	Vendor	Description
Python	Runtime	3.9 + 3.12.11	PSF	Python Software Foundation	Core runtime for all SDK and ML components.
OpenJDK	Runtime	11.0.29	GPL v2 + Classpath Exception	OpenJDK Community	Java Virtual Machine required for Apache Spark and Delta Lake.
Apache Spark (PySpark)	Runtime	3.5.5	Apache-2.0	Apache Software Foundation	Distributed compute engine for ETL, data quality and forecasting pipelines.
Delta Lake (delta-spark)	Runtime	3.0.0	Apache-2.0	Linux Foundation / Databricks Open Source	Transactional storage layer enabling ACID and time-travel on Parquet data.
pandas	Library	2.2.3	BSD-3-Clause	PyData / NumFOCUS	Core data processing library providing DataFrame structures and operations.
polars	Library	0.20.31	MIT	Polars Contributors	Fast DataFrame engine written in Rust for high-performance data manipulation.
pyarrow	Library	15.0.2	Apache-2.0	Apache Software Foundation	Columnar data access and Parquet I/O layer for Spark and Delta components.
scikit-learn	Library	1.7.2	BSD-3-Clause	Scikit-learn Developers / NumFOCUS	Machine-learning algorithms and preprocessing utilities for forecasting and modelling.
statsmodels	Library	0.14.5	BSD-3-Clause	Statsmodels Developers	Statistical modeling and time-series forecasting models such as ARIMA.
pmdarima	Library	2.0.4	MIT	Alkiviadis Management Group	Automated ARIMA model selection and time-series forecasting.
delta-spark	Library	3.0.0	Apache-2.0	Linux Foundation / Databricks	Delta Lake Python API for ACID table management and versioned data.
databricks-sql-connector	Library	3.7.1	Apache-2.0	Databricks Inc.	DBR-style SQL connector for querying Delta and Databricks tables.
fastapi	Library	0.110.0	MIT	Sebastián Ramírez	Web framework for building REST APIs and model endpoints.
uvicorn	Library	0.37.0	BSD-3-Clause	Encode OSS	ASGI server providing runtime executors for FastAPI endpoints.
azure-core / azure-identity / azure-storage-blob	Library	1.35.+	MIT	Microsoft Azure SDK	Core, authentication and blob-storage SDKs for Azure integration.
pytest / pytest-cov	Library	7.4 / 4.1	MIT	Pytest Community	Unit and integration testing framework with coverage support.
black / isort / pylint	Library	25.1 / 5.13 / 2.17	MIT / BSD-3	Python Software Foundation / PyCQA	Code formatting and linting for CI quality checks.
mkdocs / mkdocs-material / mkdocstrings	Library	1.6 / 9.5 / 0.25	MIT	MkDocs Community	Documentation generation system for SDK and forecasting components.
xgboost	Library	2.1.4	Apache-2.0	Hyunsu Cho / Jiaming Yuan / XGBoost Dev Team	High-performance gradient boosting library for regression, classification und Ranking-Tasks

Tools

Name	Category	Version	Description
git	Development / Publish	2.43.0	Distributed version control system used for source code management and collaborative development.
VS Code	Development	1.105.1	Integrated development environment used for coding, debugging, documentation editing, and Git integration.
Conda	Development / Build	25.5.1	Environment and package management system ensuring reproducible builds and dependency control.

Last Name	First Name	Value					
Böhm	**Luca**			#DIV/0!	#DIV/0!		
Meusling	**Patrick**						
Arifin	**Hafidz**						
Huy	**Christoph**						
Selbig	**Simon**			0	No size		
Haseeb	**Abdul**			1	Trivial size		
Pohnke	**Hannes**			2	Small size		
Jokiel	**Rene**			3	Medium size		
Khabouze	**Mehdi**			5	Large size		
				8	Very large size		
				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							