USER GUIDE

This document provides guidance for evaluating the Knowledge-Based Digital Twin (KBDT) prototype, designed to detect and diagnose sensor faults in wind turbine generators. Developed to support condition monitoring in wind farms, the system is undergoing incremental validation in terms of functionality, reliability, performance, and real-world applicability. It aids maintenance teams by converting sensor data—particularly from encoders and tachometers—into diagnostic insights using interactive visualizations and rule-based interpretations grounded in simulation and technical literature.

Currently, the system relies on simulated datasets generated via dynamic modeling to train machine learning and rule-based models. Future stages will include real SCADA data with injected faults to enhance validation. During this process, domain experts are expected to evaluate the reliability of statistical analyses, the practical relevance of diagnostic outputs, and the system's consistency with real turbine behavior. Comments regarding the system are placed inside blank boxes along the validation process on the web application.

SUMMARY

1. SYSTEM OVERVIEW

This section introduces the condition monitoring system developed under the digital twin paradigm. The system was built following a recursive knowledge engineering methodology, where each phase is developed incrementally. After each cycle, the outcomes are validated by a domain expert.

Two fundamental components are required to implement digital twins for condition monitoring: real operational data and a validated virtual model. These components were developed as follows:

- Model validation: A virtual wind turbine was developed using state-space equations and implemented within a multi-domain dynamic simulation platform. The model was calibrated and validated using real operational data, ensuring that it reproduces the turbine's behavior with high fidelity and reflects the physical dynamics of the real system.
- Knowledge extraction: Once the dynamic model was validated, operational data under varying wind conditions were analyzed, including scenarios with and without sensor faults. This analysis supported the construction of the digital twin’s knowledge base, forming the foundation for a rule-based inference engine capable of reasoning about system states and classifying potential faults.

This initial validation stage is essential for understanding the turbine’s behavior in a controlled environment under a range of faulty and healthy operational conditions, before deploying the system with real-world datasets. It allows for careful observation of system responses and ensures that the inference mechanisms are working as well.

As a first test case, five faults affecting encoders and tachometers were introduced into the simulation. These fault scenarios were used to populate the knowledge base and guide the construction of rule sets for the diagnostic system. A machine learning algorithm was also employed to verify the rule chaining performance and reinforce the system’s generalization capabilities.

1.1. USER’S PROFILE

This system is designed for professionals involved in wind turbine performance and/or maintenance management. It aims to support users with varying degrees of expertise, particularly those engaged in:

- Maintenance management and performance monitoring of wind turbines;
- Condition-based and predictive maintenance;
- Asset management and KPI tracking;
- Wind turbine diagnostics and operations;
- Digital transformation initiatives (Industry 4.0) within the energy sector.

Data analysts and data scientists with strong background in statistical and artificial intelligence are also welcome to test the system and recommend improvements, since they also work in the wind energy sector.

Additionally, professionals working with dynamic modeling of wind turbines, data analysts and data scientists with strong backgrounds in statistics, artificial intelligence, or machine learning are encouraged to explore the system and contribute to its improvement, particularly in the context of wind energy applications. As the prototype relies heavily on sensors like encoders and tachometers, professionals with expertise in sensor technologies are also encouraged to contribute their insights.

Although the system has been designed to be user-friendly, it assumes a basic familiarity with the operational context and turbine performance metrics. Users are expected to understand the operational data available within the system, including sensor readings, performance parameters, and failure indicators. A basic understanding of wind turbine failure modes and maintenance workflows is beneficial for making informed use of the results.

The system includes statistical visualizations, a confusion matrix, and performance scores. Users should be able to interpret classification outcomes and statistical summaries, particularly in the context of diagnostics. While the interface provides guidance and tooltips, the interpretation of advanced features — such as classification results from machine learning models or rule-based diagnostics — may require familiarity with key performance indicators and diagnostic workflows.

This documentation includes detailed instructions and background to help both newcomers and experienced maintenance managers effectively interpret system outputs and use them in support of decision-making.

2. SYSTEM TESTS

2.1. HOW TO ACCESS

The prototype of the conditional monitoring system can be accessed through the following link:

https://wt-kbdt.streamlit.app/

Streamlit is an open-source Python library that makes creating and sharing custom web applications for data analysis and machine learning easy. Although the library accepts aesthetic parameters of its functionalities through HTML, the code was developed 100% in Python, making the program expandable to any audience with basic knowledge of this programming language. The user does not need to install distributions or software that runs the Python programming language. Figure 1 shows a diagram of how the user-interface-data interaction works.

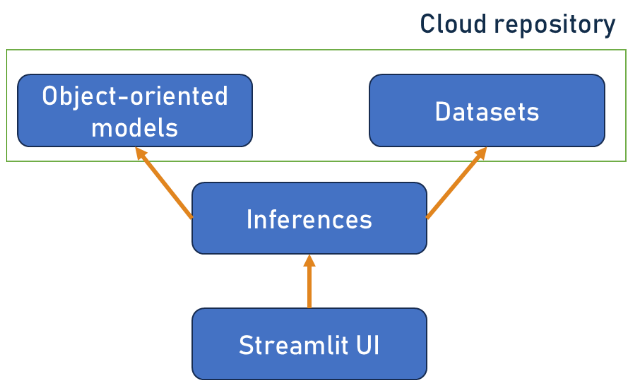
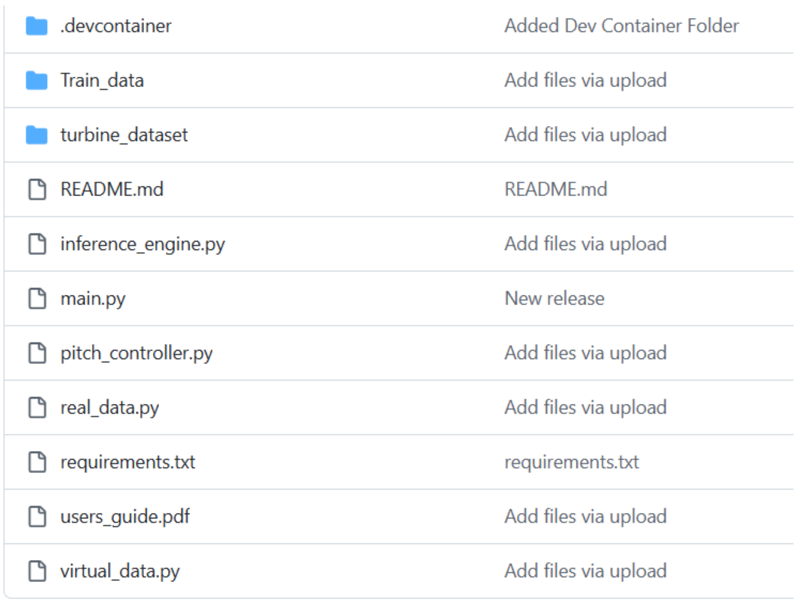


Figure 1. User interface (UI) with the prototype.

As illustrated in Figure 1, upon accessing the link, the user is directly connected to the prototype interface, which runs entirely in the cloud using Streamlit's hosted environment. All inferences and data processing occur on Streamlit's cloud infrastructure. The operational datasets and sensitivity analysis files are stored in a cloud repository and accessed dynamically by the app. Likewise, the application interface and code logic are also hosted in the same cloud environment. The organization of the files and data sources that support the interaction between the interface, processing logic, and cloud infrastructure is presented in Figure 2.



.devcontainer	Added Dev Container Folder
Train_data	Add files via upload
turbine_dataset	Add files via upload
README.md	README.md
inference_engine.py	Add files via upload
main.py	New release
pitch_controller.py	Add files via upload
real_data.py	Add files via upload
requirements.txt	requirements.txt
users_guide.pdf	Add files via upload
virtual_data.py	Add files via upload

Figure 2. Files and folders on the repository.

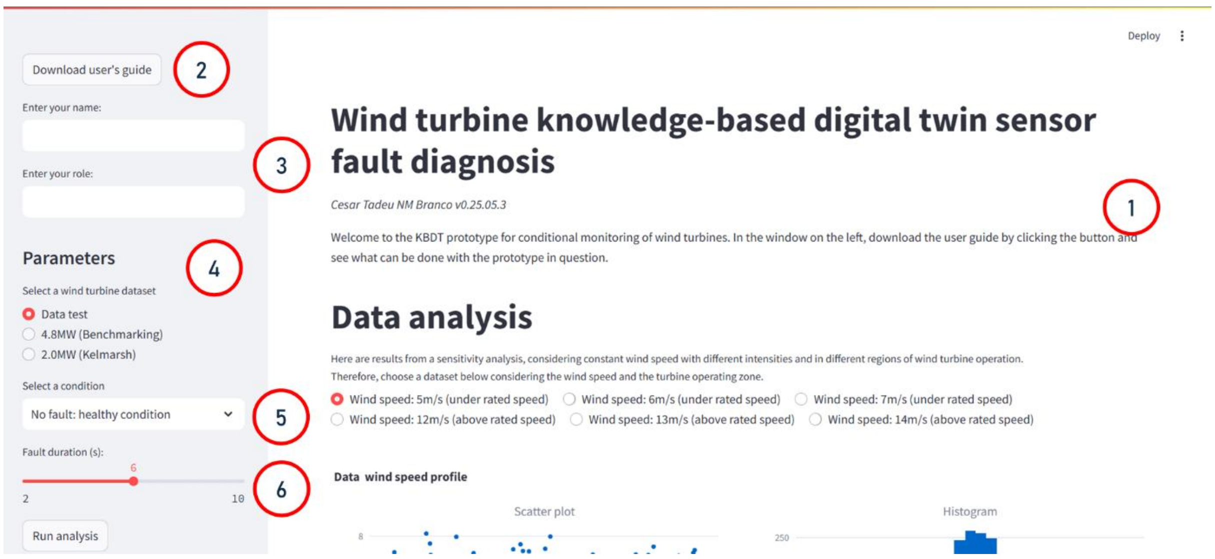
Figure 2 shows the folders and files in python format used to develop the prototype, and each element's description are available below:

- Train_data and turbine_dataset: Contains the datasets from the sensitivity analysis and the turbine's operational data.
- Pitch_controller.py: File that reads and processes the operational data from the wind turbine sensors.
- Real_data.py: File that contains functions for plotting sensor data.
- virtual_data.py: File that reads the data from the dynamic modeling and presents artificial intelligence tasks for detecting anomalies in wind turbines.
- Main.py: Main file that connects the previous files and controls the plotting of information in the interface.
- Inference_engine.py: Contains the inference system by rules in the sensors and the model for verifying the rules via machine learning.

The files presented above rely on several Python libraries that perform the main tasks of data classification and interactive plot visualization within the system. These libraries include:

- NumPy: Used for efficient numerical operations on turbine sensor data.
- Pandas: Enables structured manipulation and filtering of the turbine's operational datasets.
- Scikit-learn: Powers the classification models used to detect anomalies or predict failure conditions in turbines based on the dataset.
- Plotly: Responsible for rendering the interactive plots in the user interface, allowing users to explore turbine behavior and model outputs visually.

These libraries work together to ensure that the prototype delivers both robust data processing and clear, actionable visual insights. Having established this foundation, once the user accesses the provided link, they are presented with Figure 3.



Download user's guide

Enter your name:

Enter your role:

Parameters

Select a wind turbine dataset

[✓ Data test]
4.8MW (Benchmarking)
2.0MW (Kelmarsh)

Select a condition

No fault: healthy condition

Fault duration (s): 6

Run analysis

Deploy

Wind turbine knowledge-based digital twin sensor fault diagnosis

Cesar Tadeu NM Branco v0.25.05.3

Welcome to the KBDT prototype for conditional monitoring of wind turbines. In the window on the left, download the user guide by clicking the button and see what can be done with the prototype in question.

Data analysis

Here are results from a sensitivity analysis, considering constant wind speed with different intensities and in different regions of wind turbine operation. Therefore, choose a dataset below considering the wind speed and the turbine operating zone.

[✓ Wind speed: 5m/s (under rated speed)] Wind speed: 6m/s (under rated speed) Wind speed: 7m/s (under rated speed)
Wind speed: 12m/s (above rated speed) Wind speed: 13m/s (above rated speed) Wind speed: 14m/s (above rated speed)

Data wind speed profile

Figure 3. Initial page of the prototype.

Figure 3 presents the initial screen of the prototype. On the left, there is a sidebar containing various options, and on the right, the main sliding analysis window. This section provides a detailed overview of each screen element, with particular emphasis on the buttons relevant to the current validation cycle.

Item 1 refers to the main sliding analysis window. Here, users will find the statistical analysis of the data and the sensor fault diagnostic system. Items 2 to 6 correspond to elements of the sidebar. In 2, the user can download the User Guide document. In 3, there is a field where users can name a ".csv" file that will store their feedback and diagnostic scores as they interact with the prototype. Throughout the prototype, there are blank fields followed by reflective questions designed to guide experts through the validation process.

In 4, users can select among the available datasets used as input for the prototype. By default, the system selects “Data test,” which is derived from the sensitivity analysis study and is the focus of the first validation cycle. The datasets “4.8MW (Benchmarking)” and “2.0MW (Kelmarsh)” are real-world datasets intended for use in subsequent analysis cycles.

Finally, items 5 and 6 relate to fault injection into the real datasets (4.8MW and 2.0MW) for further analysis. These options are not relevant for the current cycle and may be disregarded at this stage. To streamline the presentation of test scenarios, the sidebar will be excluded from subsequent figures, allowing for greater clarity and emphasis on the core analytical components of the prototype.

2.2. FAULT INJECTION SETUP

To validate the performance of the rule-based diagnostic engine, specific faults were synthetically injected into the system's sensitivity analysis datasets. These scenarios simulate typical sensor anomalies that might occur in real-world wind turbine operations, allowing the expert to evaluate the system’s detection thresholds, logic, and robustness under controlled conditions. The faults were strategically introduced in both encoder and tachometer signals and span different types of abnormal behavior, such as fixed values, gain errors, and signal drifts.

The injected faults are summarized as follows:

- Fixed pitch angle (at blade A): a constant pitch value of 5° was applied from 2000s to 2100s, simulating a sensor stuck at a fixed reading.
- Gain amplification (at blade B): the encoder signal was artificially amplified by 20% between 2300s and 2400s, representing a miscalibration scenario.
- Increasing drift (at blade C): a gradual increase was added to the pitch angle from 2600s to 2700s, mimicking a sensor with drift over time.
- Fixed speed value (at rotor tachometer): the rotor speed was fixed at 13.4 rpm from 1500s to 1600s, representing a frozen tachometer reading.
- Gain amplification (at rotor tachometer): a 20% gain was applied to the tachometer signal from 400s to 500s, introducing a bias in rotational speed measurements.

These scenarios are used throughout the next test sections to calibrate diagnostic thresholds, fine-tune the system’s sensitivity, and evaluate precision-recall trade-offs. The goal is to simulate realistic failure modes and assess whether the rule-based classifier can effectively identify them under varying parameter configurations.

2.3. TEST SCENARIOS

The tests provide support to the expert in decision-making, delivering statistical analyses, heuristic information, and metrics to assess the reliability of the diagnosis.

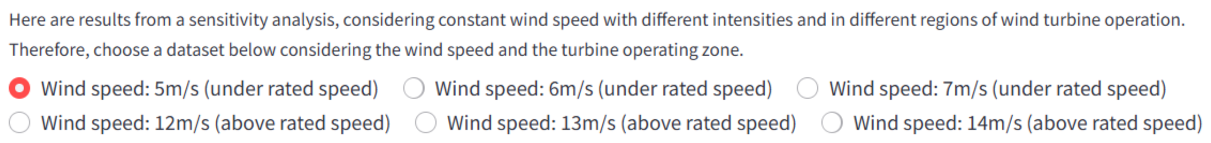
One of the innovative features of the digital twin framework is the automatic generation of expert-level insights, referred to as “Expert Digital Twin comments”. These are textual interpretations generated based on the patterns identified in the plots. They translate technical observations into actionable insights, such as:

- Deviations from normal ranges
- Patterns suggestive of misconfiguration
- Sensor alignment or synchronization issues

2.3.1. Data analysis

The data analysis of the sensitivity analysis datasets serves as a strategic foundation for validating the system’s diagnostic capabilities. It provides a visual and quantitative baseline for assessing whether the digital twin realistically simulates turbine dynamics and identifies sensor-related anomalies such as encoder drift or tachometer failure. By doing so, it supports the validation of automated inferences and expert comments generated by the system, ensuring they are consistent with field expectations and operational experience.

To enable this evaluation, the system allows users to explore multiple datasets representing distinct wind intensity profiles, each designed to expose the turbine to different operational regimes and fault conditions. These can be selected directly on the main interface, as shown in Figure 4.

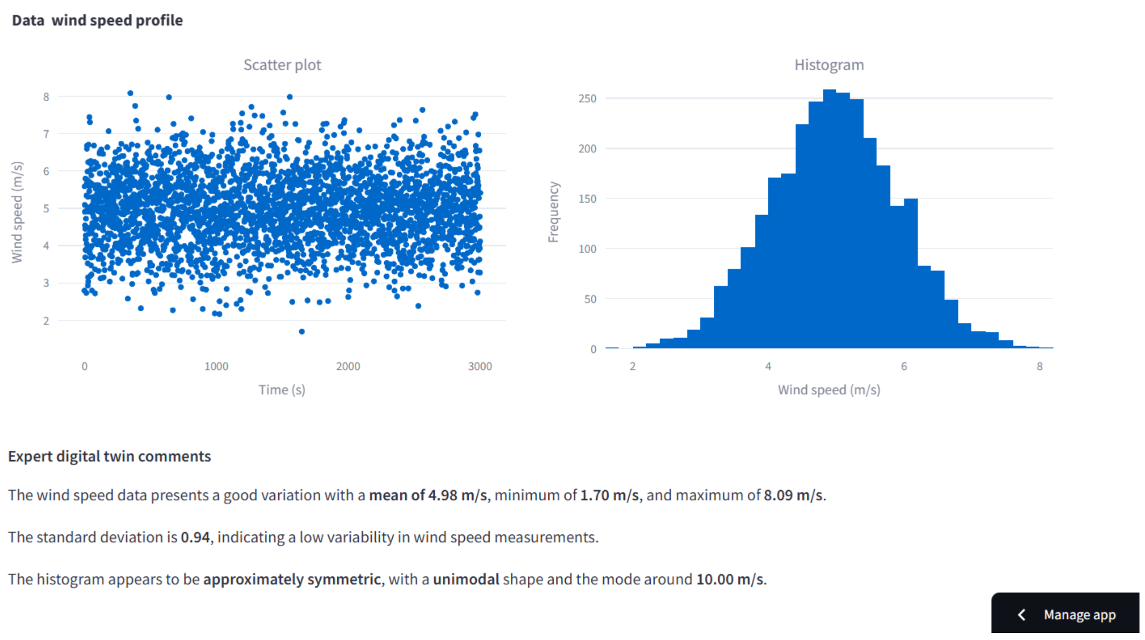

Here are results from a sensitivity analysis, considering constant wind speed with different intensities and in different regions of wind turbine operation. Therefore, choose a dataset below considering the wind speed and the turbine operating zone.

[✓ Wind speed: 5m/s (under rated speed)] Wind speed: 6m/s (under rated speed) Wind speed: 7m/s (under rated speed)
Wind speed: 12m/s (above rated speed) Wind speed: 13m/s (above rated speed) Wind speed: 14m/s (above rated speed)

Figure 4. Different datasets from sensitivity analysis that can be instantiated in the conditional monitoring system.

To assess the behavior of the wind turbine under both normal and abnormal operating conditions, a set of statistical analyses was conducted using the main variables monitored by the system: wind speed, power generation, blade pitch angles, rotor speed, and generator speed. These analyses aim to reveal underlying trends, detect non-standard patterns, and provide insight into how the turbine behaves under different operational regimes.

The tools employed for this analysis included histograms, scatter plots, boxplots, and descriptive statistics. For wind speed, a histogram and scatter plot were used to evaluate the frequency and spread of recorded values, available in Figure 5.


Data wind speed profile

Expert digital twin comments

The wind speed data presents a good variation with a **mean of 4.98 m/s**, minimum of **1.70 m/s**, and maximum of **8.09 m/s**.

The standard deviation is **0.94**, indicating a low variability in wind speed measurements.

The histogram appears to be **approximately symmetric**, with a **unimodal** shape and the mode around **10.00 m/s**.

Figure 5. Wind speed scatter plot and histogram.

This information serves as a baseline to understand how the turbine reacts under different wind intensities and to identify operating thresholds that may influence system dynamics or lead to faulty behavior. However, our main focus lies on sensor. Therefore, encoders for the three blades are displayed as scatter plots and boxplots (Figure 6).

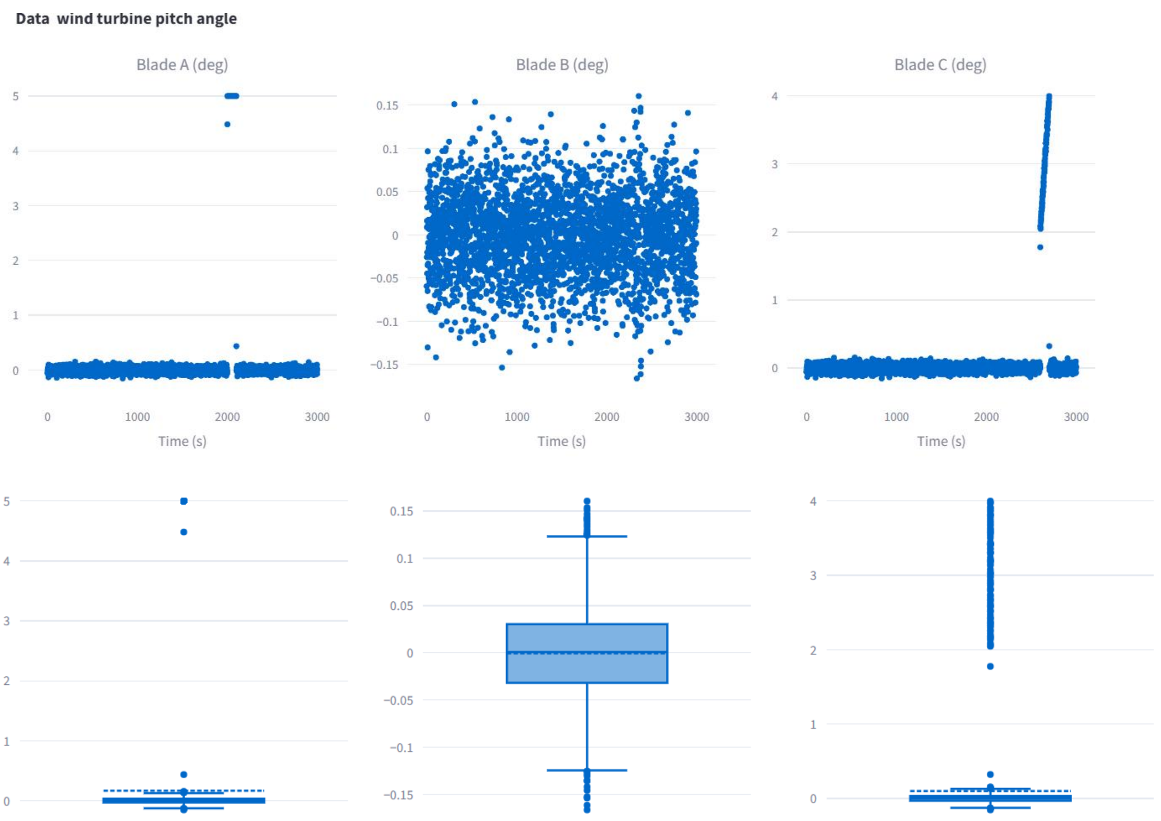
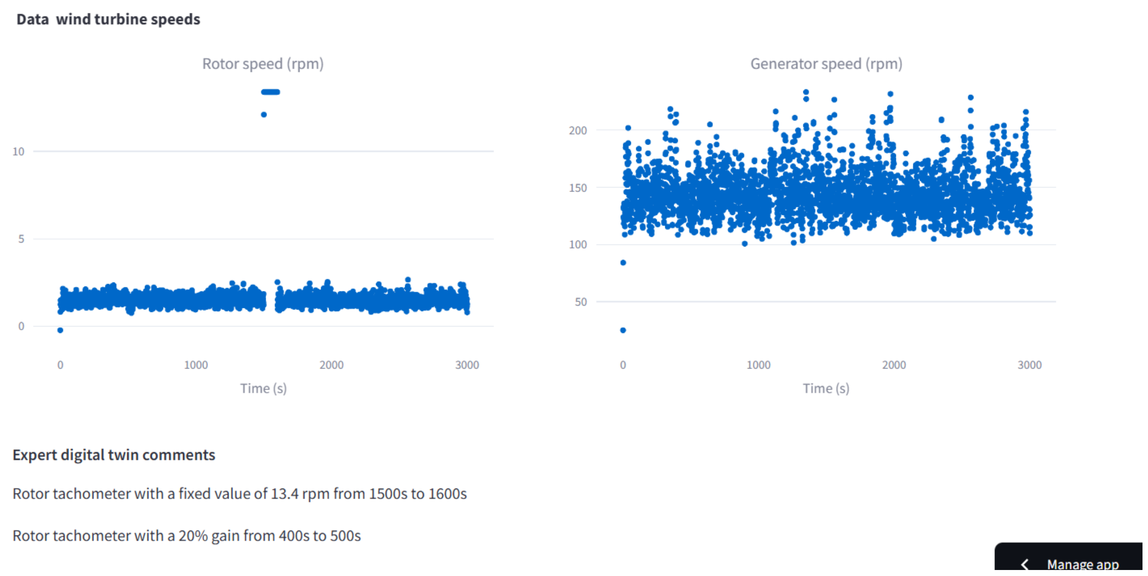


Figure 6. Scatter and boxplot of the blades.

Some points appear to be out of place in the scatter plot. These points are precisely faults injected at specific moments in the simulation. Details of the faults are specified in the prototype. This analysis already gives an idea of whether or not there is an anomaly in the blade sensor. The average angle of each blade is inside the boxes, followed by a comment from the "Expert Digital Twin". Furthermore, another scatter plot for rotor and generator tachometers are stated according to Figure 7.



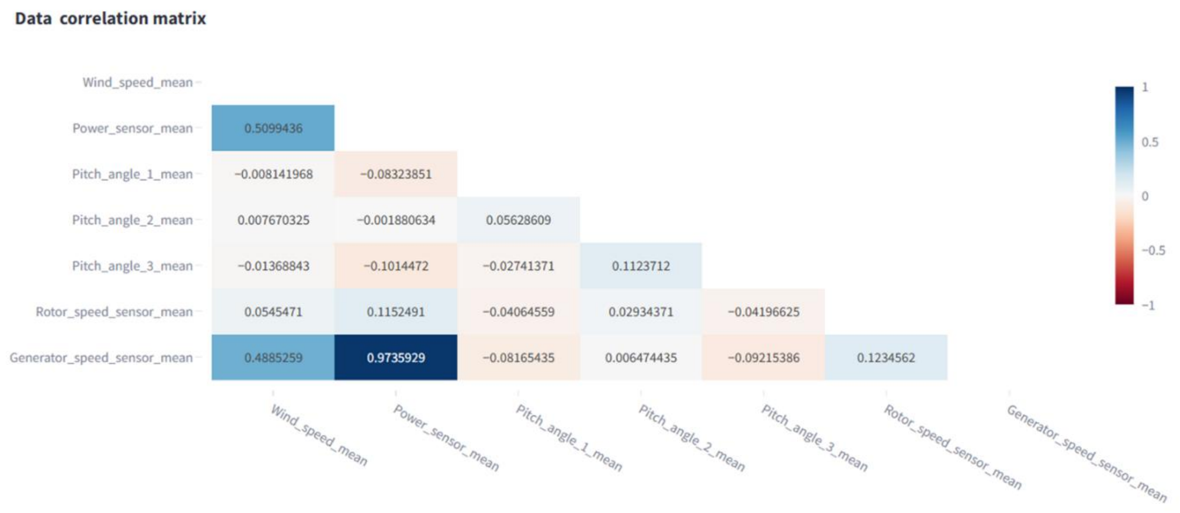
Expert digital twin comments

Rotor tachometer with a fixed value of 13.4 rpm from 1500s to 1600s

Rotor tachometer with a 20% gain from 400s to 500s

Figure 7. Rotor and generator speed scatter plots.

Again, there are extreme points that represent faults in the tachometers, in addition to the comments from the “Expert Digital Twin”. The fault descriptions are presented in the prototype. To synthesize the relationships among all monitored variables, a correlation matrix was constructed according to Figure 8.



	Wind_speed_mean	Power_sensor_mean	Pitch_angle_1_mean	Pitch_angle_2_mean	Pitch_angle_3_mean	Rotor_speed_sensor_mean	Generator_speed_sensor_mean
Wind_speed_mean							
Power_sensor_mean	0.5099436						
Pitch_angle_1_mean	−0.008141968	−0.08323851					
Pitch_angle_2_mean	0.007670325	−0.001880634	0.05628609				
Pitch_angle_3_mean	−0.01368843	−0.1014472	−0.02741371	0.1123712			
Rotor_speed_sensor_mean	0.0545471	0.1152491	−0.04064559	0.02934371	−0.04196625		
Generator_speed_sensor_mean	0.4885259	0.9735929	−0.08165435	0.006474435	−0.09215386	0.1234562	

Figure 8. Correlation matrix among all variables.

This Figure 8 provides a high-level overview of how variables interact, helping to detect redundancies, strong dependencies, particularly useful for refining the knowledge base and improving fault inference logic.

2.3.1.1. Data analysis qualitative assessment

At the end of the statistical analysis, the expert is invited to leave his impressions based on a text to guide his writing.

Do the statistical analyses and comments reflect the real operational behavior of wind turbines? Do the simulated faults in encoders and tachometers resemble current failure patterns observed in wind farms? Please elaborate on your assessment.
Expert comments:

2.3.2. Threshold tuning for rule-based diagnosis

A robust diagnostic system depends critically on the careful calibration of thresholds that define when an alert should be triggered. These thresholds must find a balance, strict enough to detect actual faults (minimizing false negatives), but not so sensitive that they misclassify normal operational variance as failure (minimizing false positives). This balance is quantitatively represented by the trade-off between precision and recall, two key performance metrics in fault detection systems.

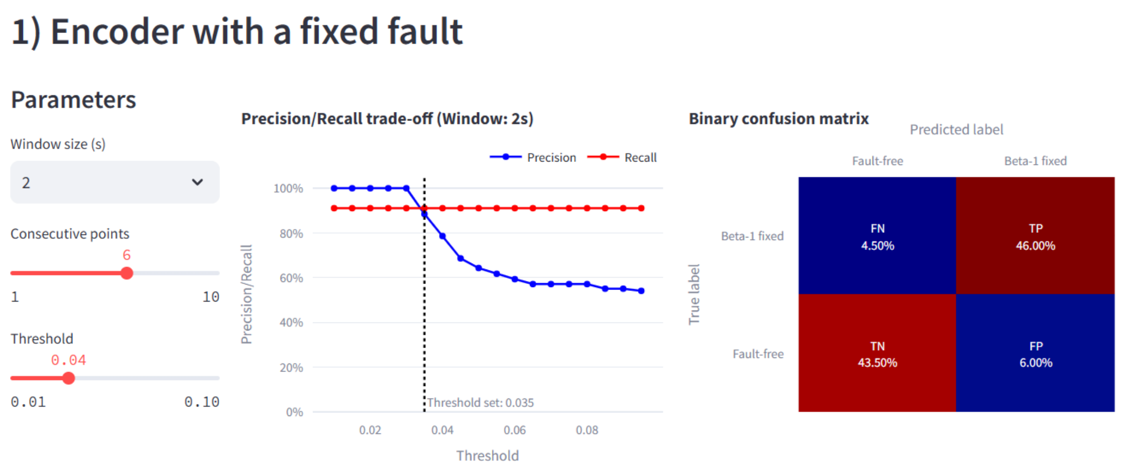
In this test scenario, the expert is provided with the ability to calibrate threshold parameters for individual failure cases involving both the blade pitch encoders and the rotor tachometer. The available parameters include:

- Window size: the temporal resolution used to analyze the data,
- Number of consecutive points: used to confirm the persistence of the anomaly,
- Threshold value itself: the magnitude beyond which an alert is triggered.

These parameters setup a rule-based inference system that evaluates sensor data and displays binary fault classifications. The resulting binary classification performance is visualized in two complementary ways:

- Precision/Recall tradeoff plots, illustrating the trade-offs associated with each parameter configuration. Hint: try to hover the mouse on the plot to see the percentage of precision and recall varying along different thresholds.
- Binary confusion matrices, offering a direct view into the percentage of true positives, false positives, true negatives, and false negatives. Here the samples are purposefully balanced, i.e., there is a 50/50 ratio between healthy and faulty samples. Hover on it and you will see the samples classified inside each class, instead of percentage values.

As an example of this section trade-off system, Figure 9 presents an illustration of how the sensitivity analysis of the detection system can be performed.



1) Encoder with a fixed fault

Parameters

Window size (s): 2

Consecutive points: 6 (1 – 10)

Threshold: 0.04 (0.01 – 0.10)

Binary confusion matrix

True label \ Predicted label	Fault-free	Beta-1 fixed
Beta-1 fixed	FN 4.50%	TP 46.00%
Fault-free	TN 43.50%	FP 6.00%

Figure 9. Encoder with a fixed fault.

Figure 9 presents the performance of the rule-based diagnostic system when applied to a scenario where the pitch angle A is locked at 5° between 2000s and 2100s. The tuning parameters directly affect the trade-off between early detection and false alarms.

On the left, the user adjusted a 2-second window with a requirement of 6 consecutive points and a 0.04 threshold. The Precision/Recall plot shows that while recall remains high

(~90%), precision drops notably as the threshold decreases, signaling a potential increase in false alarms in field conditions.

- The confusion matrix on the right helps contextualize this for operational maintenance:
- True positives (46%) indicate the proportion of successful fault detections during the faulty interval.
- False positives (6%) represent periods where the system flagged faults in normal operation, which could translate into unnecessary inspections or maintenance interventions in real wind farms.
- False negatives (4.5%) suggest missed faults, a critical metric for safety-critical systems.

These examples demonstrate how each fault signature influences the system's sensitivity and diagnostic consistency. By interacting with each scenario, domain experts are expected to reflect on the practical implications of each threshold configuration, and whether these rules could be trusted to support reliable and scalable maintenance planning in operational wind farms.

2.3.2.1. Threshold tuning qualitative assessment

At the end of the threshold tuning stage, the expert is invited to leave his impressions based on a text to guide his writing.

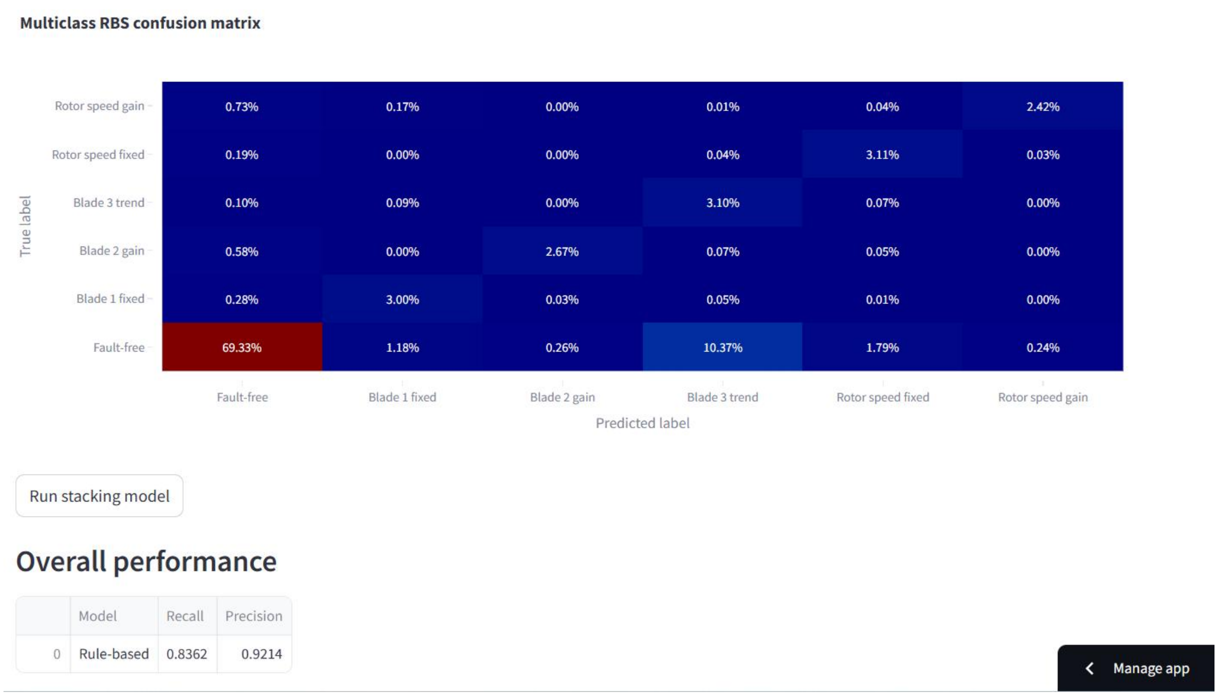
Do the threshold settings and binary alerts reflect realistic maintenance scenarios in wind turbines? Would these parameter configurations support accurate and timely fault identification in a real operational environment? Are the monitored variables (e.g., pitch angle, rotor speed) suitable for the type of failures being diagnosed? Please elaborate on your assessment.
Expert comments:

2.3.3. Single-shot multiclass prediction

In this phase of the system evaluation, the multiclass classification analysis is presented as a realistic scenario where different fault types may occur at any time during turbine operation. Unlike the binary analysis, which isolates one fault at a time, the multiclass approach consolidates all fault types and healthy cases into a single dataset. The rule-based system is then applied to this combined dataset, replicating a real-world diagnostic workflow where a single algorithm must scan operational data and classify the presence and type of any fault, if any.

All the sensitivity analysis datasets have been concatenated, and a single-shot classification was applied. This means the system had one chance to evaluate the full dataset and return the most probable fault class (or identify a fault-free condition) for each instance. This setup aims to reflect how diagnostic systems behave in live monitoring environments, where the algorithm must continuously analyze incoming data and respond dynamically.

Figure 10 shows the confusion matrix generated from this multiclass test, where each row represents the true condition of the turbine and each column reflects the system's predicted classification.



Multiclass RBS confusion matrix

True label \ Predicted label	Fault-free	Blade 1 fixed	Blade 2 gain	Blade 3 trend	Rotor speed fixed	Rotor speed gain
Rotor speed gain	0.73%	0.17%	0.00%	0.01%	0.04%	2.42%
Rotor speed fixed	0.19%	0.00%	0.00%	0.04%	3.11%	0.03%
Blade 3 trend	0.10%	0.09%	0.00%	3.10%	0.07%	0.00%
Blade 2 gain	0.58%	0.00%	2.67%	0.07%	0.05%	0.00%
Blade 1 fixed	0.28%	3.00%	0.03%	0.05%	0.01%	0.00%
Fault-free	69.33%	1.18%	0.26%	10.37%	1.79%	0.24%

Overall performance

	Model	Recall	Precision
0	Rule-based	0.8362	0.9214

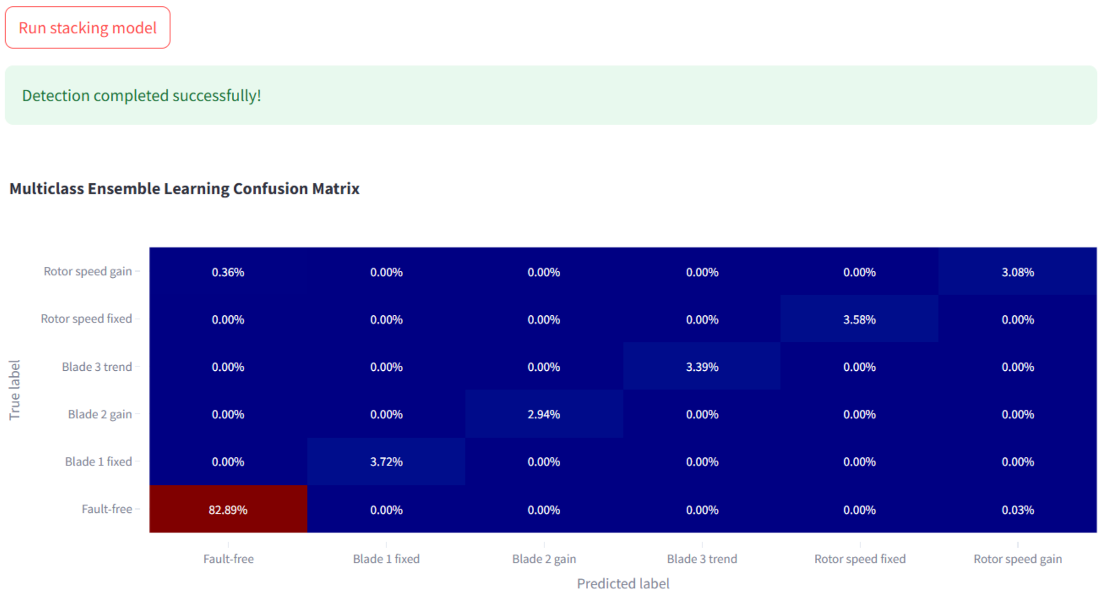
Figure 10. Confusion matrix and the metrics for the rule-based system.

As shown in Figure 10, each cell represents the percentage of classifications that fall into a specific true-predicted class combination. The diagonal cells correspond to correct classifications (true positives), while the off-diagonal cells represent various types of misclassifications. Users are encouraged to focus on the diagonal to assess the system's effectiveness in correctly identifying each fault type. The system's overall recall (83.6%) and precision (92.1%) for this multiclass setup are provided below the matrix and should be interpreted in the context of the imbalance present in the dataset, where healthy conditions are more frequent than faulty ones.

The dataset used contains approximately 18,000 instances. This is a relevant detail for the evaluator, as the percentage values shown in the matrix can be converted into absolute counts to understand how many individual cases were handled correctly or incorrectly. For example, a 3% true positive rate for a specific fault class would represent around 540 correctly classified instances. Understanding this scale helps to appreciate the operational relevance of each fault detection and supports evaluating whether the system could, in practice, reduce unnecessary inspections, prevent downtime, or guide timely interventions.

Experts are encouraged to interpret these results considering their own experience with turbine operation and failure diagnostics. For example, identifying a fixed blade fault correctly, even 3% of the time in a large dataset, could represent dozens or hundreds of potential interventions avoided or better targeted.

To further validate the robustness of the rule-based inference engine, an ensemble learning model was applied. This model aggregates the predictions of multiple optimized classifiers into a final output, improving fault detection by leveraging complementary strengths of individual algorithms. Although several machine learning models were previously trained and tuned offline, only the top-performing ensemble configuration was embedded into the system for runtime efficiency. Users can execute this model by clicking the "Run stacking model" button in the interface, as stated in Figure 11.



Detection completed successfully!

Multiclass Ensemble Learning Confusion Matrix

True label \ Predicted label	Fault-free	Blade 1 fixed	Blade 2 gain	Blade 3 trend	Rotor speed fixed	Rotor speed gain
Rotor speed gain	0.36%	0.00%	0.00%	0.00%	0.00%	3.08%
Rotor speed fixed	0.00%	0.00%	0.00%	0.00%	3.58%	0.00%
Blade 3 trend	0.00%	0.00%	0.00%	3.39%	0.00%	0.00%
Blade 2 gain	0.00%	0.00%	2.94%	0.00%	0.00%	0.00%
Blade 1 fixed	0.00%	3.72%	0.00%	0.00%	0.00%	0.00%
Fault-free	82.89%	0.00%	0.00%	0.00%	0.00%	0.03%

Figure 11. Confusion matrix for stacking machine learning algorithm.

The resulting confusion matrix, shown in Figure 11, provides a comparative reference to assess how the ensemble performs relative to the rule-based system. Experts should examine this matrix carefully, comparing class-level performance and overall precision-recall dynamics to identify strengths, blind spots, or patterns of confusion unique to either model. If certain faults consistently result in low classification rates or are confused with others, this may warrant a deeper review of rule conditions, input signal clarity, or even sensor placement and quality.

The next section invites the expert to reflect on the system's output from both technical and operational standpoints. Their feedback is crucial in identifying practical shortcomings and in determining whether the current version of the system is sufficiently mature for deployment in real-world turbine monitoring.

2.3.3.1. Multiclass prediction qualitative assessment

Experts are now invited to critically assess the performance and operational utility of the multiclass classification approach. Please consider both the rule-based and ensemble-based confusion matrices in your evaluation, reflecting on their diagnostic value, reliability, and relevance to real-world turbine maintenance scenarios.

Does the system's classification performance, as seen in the confusion matrices, provide an accurate and operationally useful representation of fault and healthy conditions? How do you interpret the trade-offs between detection rates and misclassifications across different fault types? In your experience, do the input variables and rule logic offer sufficient discriminatory power for real-time diagnostics? Would you recommend any changes in variable selection, rule thresholds, or system logic to improve fault differentiation and reduce diagnostic ambiguity?
Expert comments:

3. QUANTITATIVE EVALUATION

At the end of each validation test, users are invited to provide qualitative feedback by answering open-ended questions tailored to each functionality of the system. These responses are essential to understanding the perceived value and practical relevance of the prototype from an expert's perspective.

In parallel, a quantitative assessment is also performed through a star rating system (from 1 to 5 stars) across the following ten core attributes:

Effectiveness of classification output: evaluates whether the system is correctly identifying and classifying faults, which is central to validating its diagnostic reliability.

Similarity of simulated failures to real-world conditions: checks how realistic and representative the simulated faults are when compared to actual failures observed in operational wind farms.

Clarity of diagnostic comments and visualizations: measures how clearly the system communicates its findings through automated comments and interactive visualizations.

Relevance of thresholds and parameters for diagnostics: assesses whether the thresholds and decision rules used in both rule-based and ML-driven diagnostics make sense in real-world applications.

Ease of navigation and interface clarity: Captures the user experience in terms of how intuitive and accessible the system interface is.

Usefulness for maintenance planning and prioritization: determines whether the outputs of the system effectively support planning and prioritizing maintenance actions.

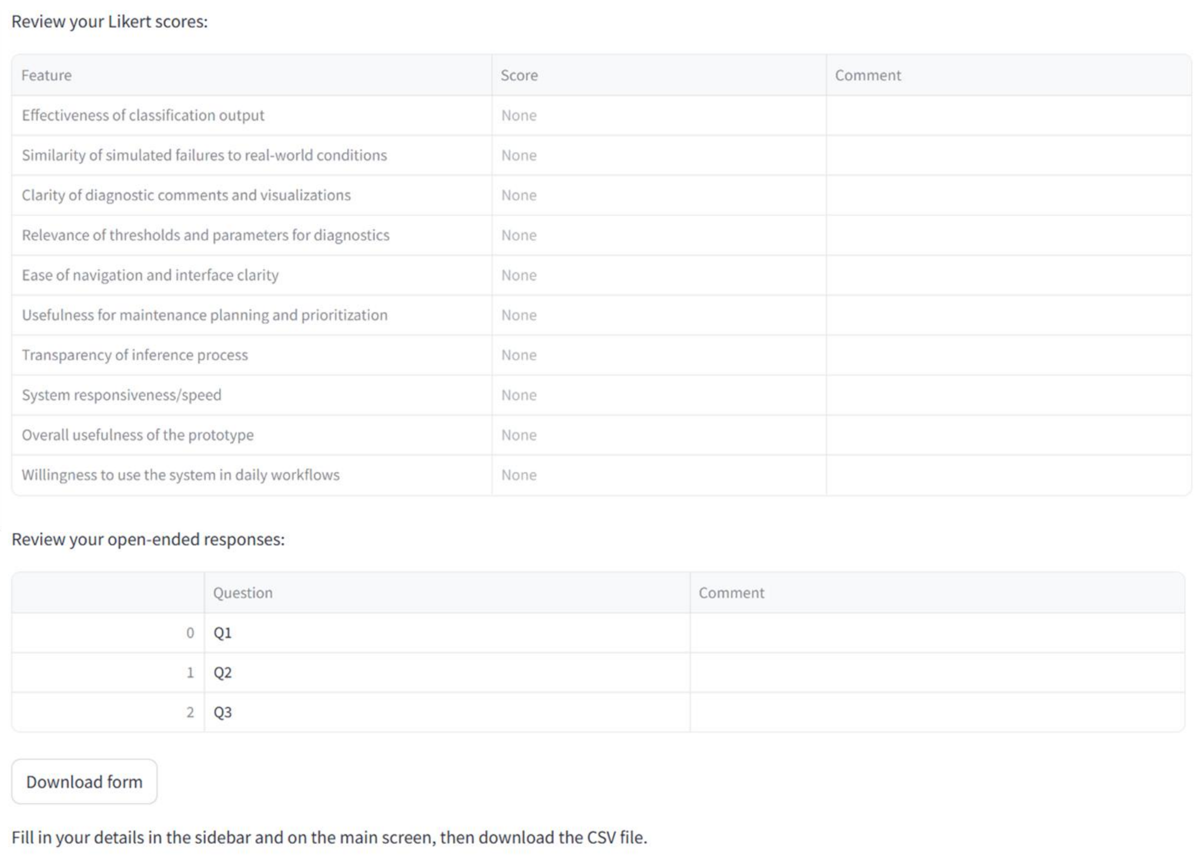
Transparency of inference process: evaluates how well the system explains the reasoning behind its conclusions, contributing to trust in its outputs.

System responsiveness/speed: assesses how quickly the system responds to user interactions and processes data, ensuring a smooth experience.

Overall usefulness of the prototype: assesses the overall value and contribution of the tool as a support system for condition monitoring and maintenance decision-making.

Willingness to use the system in daily workflows: measures the user's intention or likelihood to adopt the system in their routine work processes.

After evaluate the entire system, the user reaches a similar screen proposed on Figure 12.


Review your Likert scores:

Feature	Score	Comment
Effectiveness of classification output	None	
Similarity of simulated failures to real-world conditions	None	
Clarity of diagnostic comments and visualizations	None	
Relevance of thresholds and parameters for diagnostics	None	
Ease of navigation and interface clarity	None	
Usefulness for maintenance planning and prioritization	None	
Transparency of inference process	None	
System responsiveness/speed	None	
Overall usefulness of the prototype	None	
Willingness to use the system in daily workflows	None	

Review your open-ended responses:

	Question	Comment
0	Q1	
1	Q2	
2	Q3	

Fill in your details in the sidebar and on the main screen, then download the CSV file.

Figure 12. Review comments and scores screen and download form.

Figure 12 depicts the summary screen where all star ratings and qualitative comments are displayed for review. Users can revise and confirm their responses. Upon confirmation, the user is prompted to click “Download form”, which generates a downloadable Excel file containing all responses. If the user has provided their name and role, the file will be named in the format: “name_role_data.csv”. Otherwise, it will be saved as “unknown_user.csv”.

Participants are kindly requested to submit this file to the following email address: engmbcesar@gmail.com. To ensure your email is successfully delivered and not mistakenly flagged as spam, please also use an alternative method (e.g., send an email to Professor Jonny or contact me at LinkedIn) to confirm that you have completed the validation process and submitted the file.