American University Kyiv

A SOFTWARE SOLUTION FOR INTELLIGENT FAULT DETECTION IN WHEEL BEARINGS

ПРОГРАМНЕ РІШЕННЯ ДЛЯ ІНТЕЛЕКТУАЛЬНОГО ВИЯВЛЕННЯ НЕСПРАВНОСТЕЙ У ПІДШИПНИКАХ КОЛІС

by Glib Glugovskiy

Presented in Partial Fulfillment of the Requirements for the Degree

Master of Software Engineering

APPROVED BY:

Jacek Leśkow

Ph.D., Rector of American University Kyiv

2025

1. ABSTRACT

Wheel bearings are critical components in machinery and vehicles, and their failure can lead to costly downtime and safety risks. This project focuses on the research, development, and implementation of an intelligent software solution for fault detection in wheel bearings. By combining advanced signal processing techniques and machine learning, the solution enables early and accurate fault diagnosis efficiently for all users.

The software incorporates Fast Fourier Transform (FFT) for feature extraction and Self-Organizing Maps (SOMs) for clustering and fault classification, achieving high accuracy in detecting bearing defects. This approach marks a significant advancement in the fields of machine maintenance and fault diagnostics by offering a more precise and efficient method for identifying defects in wheel bearings.

A key aspect of the project involves the development and training of a SOM model. This model is specifically designed to analyze patterns in the cyclostationary signal data, enabling the accurate identification of potential bearing faults. The SOM model, along with the other signal processing and machine learning components, is integrated into a robust software application. Built using Django and React.js for the web interface and enhanced with Rust libraries for computational efficiency, this solution presents a novel integration of traditional web technologies with modern performance optimization techniques.

2. ACKNOWLEDGEMENTS

I want to sincerely thank my project advisor, Prof. Dr. Jacek Leśkow, for his guidance, support, and mentorship throughout this research. His expertise and insightful feedback have significantly contributed to the quality of this paper.

3. TABLE OF CONTENTS

4. LISTS OF ABBREVIATIONS

SOM	Self-Organizing Map
PSD	Power Spectral Density
MAD	Mean Absolute Deviation
SVM	Support Vector Machine
ORM	Object-Relational Mapping

1. INTRODUCTION

The increasing complexity and automation in the automotive and machinery maintenance sectors highlight the need for reliable fault detection methodologies to ensure safety and efficiency. Wheel bearings, critical components that reduce friction between moving parts, are widely used in applications such as cars, trains, and industrial machinery. Failure in wheel bearings can lead to significant economic losses, safety hazards, and operational inefficiencies. In sectors like automotive and aerospace, undetected bearing defects pose serious safety risks with potentially catastrophic outcomes [3].

Economically, the cost of undetected bearing defects includes expensive repairs, replacement of damaged components, and downtime, which disrupts productivity and supply chains. Current diagnostic methods often require stopping machinery for manual inspections, resulting in delays and additional costs. Early detection systems are essential to mitigate these issues, enabling continuous monitoring, reducing unexpected downtimes, and ensuring both operational efficiency and safety [4].

This project explores statistical signal processing and machine learning techniques to accurately identify bearing defects and develops a practical software application to assist mechanics in diagnosing bearing issues early. The project incorporates cyclostationary signal analysis, which has proven effective in identifying distinctive defect features in rotating machinery [2], and has been used lately as a diagnostic indicator of faulty bearings, helping in the selection of the optimal frequency band for demodulation [3]. In combination with unsupervised machine learning techniques such as k-means clustering and Self-Organizing Maps, these methods have demonstrated success in complex pattern recognition tasks across various domains [6].

1.1 Relevance of the Topic

The project represents a combination of state-of-the-art technologies in software engineering and signal processing, setting a new standard in the field of machine

maintenance and fault diagnosis. This technological innovation can revolutionize how industries approach machinery maintenance, making it a significant step forward in academic research and practical application.

Furthermore, the relevance of this capstone project lies in leveraging machine learning for the early detection of wheel bearing defects to prevent costly failures and downtime. The integration of machine learning techniques in the diagnostic process offers the potential for more accurate and reliable fault detection. Unsupervised machine learning algorithms can analyze complex patterns in the data that might be overlooked by traditional methods, leading to earlier and more accurate detection of anomalies.

In addition to these advanced signal processing techniques, the project will integrate cutting-edge software technologies to build a robust and efficient diagnostic tool. The core predictive algorithms will be implemented in the Rust programming language, which is renowned for its safety, speed, and efficient memory management. Rust's capabilities make it an ideal choice for handling the high-performance computing tasks required in real-time signal analysis and fault detection.

The integration of Rust and Python for backend computation, and HTMX, and React.js for the frontend represents a significant technological advancement. It not only enhances the accuracy and efficiency of fault detection but also improves the user experience, making the tool more accessible and practical for real-world applications [7].

1.2 Literature Review

Recent advancements in fault detection for rolling bearings have introduced innovative methodologies that extend beyond traditional signal processing techniques. The study "CMAFI — Copula-based Multifeature Autocorrelation Fault Identification of rolling bearing" by Duda, Leśkow, Pawlik, and Cioch (2024) proposes a novel approach leveraging copula-based multifeature autocorrelation analysis. This method bypasses the limitations of frequency band pre-selection required in conventional envelope spectrum techniques by directly analyzing statistical dependencies in the time series through copula functions. Utilizing principal component analysis (PCA) for feature selection and Fourier

analysis for frequency characterization, the CMAFI approach effectively identifies subtle damage-related patterns, especially in non-Gaussian, heavy-tailed signals. Empirical evaluations on rolling bearings with various defects demonstrate that CMAFI outperforms envelope spectrum methods in certain scenarios, particularly for rolling element damage, highlighting its potential for automation and broader applicability in variable operational conditions [1].

The exploration of statistical and machine learning methods in the detection of faults in machine wheel bearings has been marked by significant advancements, as evidenced by extensive research efforts. A major part of research in bearing fault detection heavily focuses on statistical signal processing. Studies such as the one conducted by Cioch, Knapik, and Leskow “Finding a frequency signature for a cyclostationary signal with applications to wheel bearing diagnostics”, highlight the effectiveness of cyclostationary signal analysis in isolating characteristic features of bearing faults. This approach, which concentrates on frequency-related features, offers a more accurate diagnostic tool compared to traditional time-domain analysis [2]. Complementing this, “A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings” by Tandon and Choudhury provide a comprehensive review of methods for detecting defects in rolling element bearings, further underscoring the importance of signal processing techniques in diagnosing bearing health [4].

The study "High-speed bearing diagnostics: Observations from the Surveillance 8 Safran contest data" by Smith et al. investigates the cyclostationary properties of high-speed bearing signals, emphasizing the need to distinguish first- and second-order components for accurate fault detection. The research highlights the role of targeted signal processing techniques, such as spectral correlation and envelope spectrum analysis, and demonstrates how effective windowing and data preparation, like removing shaft-synchronous components and optimizing frequency ranges, enhance diagnostic accuracy. These approaches can be directly applied to this project to improve the precision of cyclostationary analysis and optimize fault detection workflows [3].

In the domain of machine learning, there has been a growing emphasis on unsupervised learning methods, such as K-means clustering and Self-Organizing Maps (SOMs), in bearing fault detection. The study “Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm” by Tian, Azarian,

and Pecht discusses the application of SOMs alongside a K-nearest neighbor algorithm for anomaly detection. These methods excel in processing complex data patterns and automatically categorizing them, which is crucial in identifying abnormal signal patterns indicative of bearing faults [6].

Other studies have shown that supervised learning algorithms, such as Support Vector Machines (SVMs) and Neural Networks, can be highly effective in classifying and predicting bearing faults when provided with a sufficiently large and well-labeled training dataset. These methods leverage historical data where the health status of bearings is known, allowing the algorithms to learn and identify the characteristics of both healthy and faulty bearings. However, it is important to note that the effectiveness of supervised learning is heavily dependent on the quality and quantity of the training data. Inadequate or poorly labeled data can lead to inaccurate predictions and reduced reliability of the fault detection system [8].

The integration of statistical signal processing and machine learning methods has been shown to offer a comprehensive approach to fault detection. For instance, Wodecki, Michalak, and Zimroz demonstrate how machine learning algorithms can enhance the accuracy of signal processing techniques in identifying bearing defects in their work “Optimal filter design with progressive genetic algorithm for local damage detection in rolling bearings”. This interdisciplinary approach not only improves diagnostic accuracy but also contributes significantly to operational efficiency by enabling more effective and automated fault detection processes [5].

Overall, the shift towards sophisticated, automated diagnostic tools capable of efficiently handling large data volumes and offering reliable fault detection. The merging of advanced signal processing methods with machine learning algorithms, as illustrated in these studies, marks a significant advancement in the field. This trend enhances diagnostic accuracy and contributes to safer and more efficient machinery operations.

1.3 Project’s Goal

The primary goal of this Capstone Project is to create an innovative software solution for early detection of wheel bearing defects in rotating machinery. This solution will implement advanced signal analysis techniques and machine learning

algorithms to facilitate automated diagnosis. The project encompasses designing, developing, and deploying a software tool integrating cyclostationary analysis and machine learning methods such as K-means clustering and Self-Organizing Maps (SOMs). A key aspect of the project is establishing a robust software development and deployment pipeline, which will adhere to industry-standard practices to ensure scalability, maintainability, and high performance.

This project aims to increase diagnostic capabilities through unsupervised learning algorithms, primarily focusing on K-means and SOMs. It also aims to design and implement software architecture that seamlessly integrates the signal processing and machine learning modules while adhering to best practices such as modular design and version control. Furthermore, we aim to develop a user-friendly interface that allows engineers or maintenance personnel to easily use the tool for defect diagnosis. My objective is also to create a deployment pipeline that includes cloud-based storage and processing capabilities, thereby supporting the scaling of the software application.

To achieve these objectives, the project will undertake the following specific tasks:

1. Manually inspect wheel-bearing signal datasets and create sequence diagrams for systematic inspection.
2. Conduct preliminary data analysis to pinpoint characteristics that accurately define defects.
3. Utilize the k-means algorithm for initial categorization and pattern recognition.
4. Implement and assess Self-Organizing Maps for advanced analysis and feature extraction.
5. Design the software architecture for the system, outline an implementation plan, and determine a development approach.
6. Develop the software tool, incorporating the algorithms for automated defect detection, with Rust code transpiled from MATLAB scripts.

Furthermore, the project seeks to answer the following open questions:

> What signal processing features most effectively differentiate healthy from defective wheel bearings in various conditions?

How accurately does the k-means clustering algorithm categorize signal features for different types of bearing defects?

Can Self-Organizing Maps enhance fault detection accuracy compared to other unsupervised learning techniques?

1.4 Structure of the Capstone Project

This Capstone Project is structured to provide a comprehensive understanding of the development and implementation of a software solution for the detection of wheel bearing defects in rotating machinery. The project is divided into several key sections, each focusing on a specific aspect of the project, from the algorithm exploration and development to the results and conclusions.

The Introduction sets the stage for the project, outlining its objectives, significance, and context. The Topic Relevance and Literature Review sections that follow emphasize the importance of early fault detection in wheel bearings within an industrial context. The Literature Review section explores key developments in statistical and machine learning techniques for fault detection, incorporating insights into the capstone project from recent research on cyclostationary signal processing and other modern methods for bearing diagnostics.

The Methodology section is divided into two subsections: Data Nature and Data Analysis outlines the approaches and techniques used for collecting and analyzing the data relevant to wheel bearing faults; Algorithm Implementation and Comparison Outcomes describes the implementation of various algorithms and compares their effectiveness in detecting faults.

The Software Structure section provides a more in-depth look into the architectural design of the software system. It includes:

- System Architecture, which details the decisions and technologies selected to build a solution, including the front-end built with ReactJS and its communication with a Python-based back-end through a REST API. The back-end processes input data, interacts with an ML model (with deployment options via Docker or Google ML Engine), and manages data storage in systems like Redis or PostgreSQL.

- Interface and Interaction Flow, which discusses the design of the user interface and the flow of interaction within the software, focusing on how users interact with the system for data visualization and predictions.
- System Deployment, which covers the strategies and considerations for deploying the software, including scalability and maintenance aspects.

The Results section analyzes the project outcomes, presenting the effectiveness of the implemented solution in detecting wheel bearing defects. The Conclusion summarizes the key findings and contributions of the project and suggests potential future directions for research and development. The References section lists all the sources and literature referenced throughout the project, providing a comprehensive bibliography for further reading and validation of the research.

2. METHODOLOGY

2.1 Nature of the Signal Data

As discussed above, detecting wheel bearing faults, crucial for ensuring the safety and operational efficiency of mechanical systems. Consequently, the ability to accurately identify and predict these faults is of paramount importance in the domain of mechanical diagnostics. The methodology described below leverages data collection and signal processing techniques to address this challenge, aiming to provide a reliable means of early fault detection that could substantially mitigate the risks associated with bearing failures.

The project utilizes data collected from the Laboratory of Mechanical Diagnostics at the AGH University of Science and Technology in Cracow, using an experimental setup. This setup comprises several components: an electric motor (1), a gearbox (2), and the tested wheel bearing type 1206K (3), as illustrated in Fig. 1 of the attachments. The photo with bench setup is obtained from the paper by Cioch, Knapik, and Leskow [2], which provides a detailed overview of the experimental technique for collecting wheel bearing signal for future diagnostics.

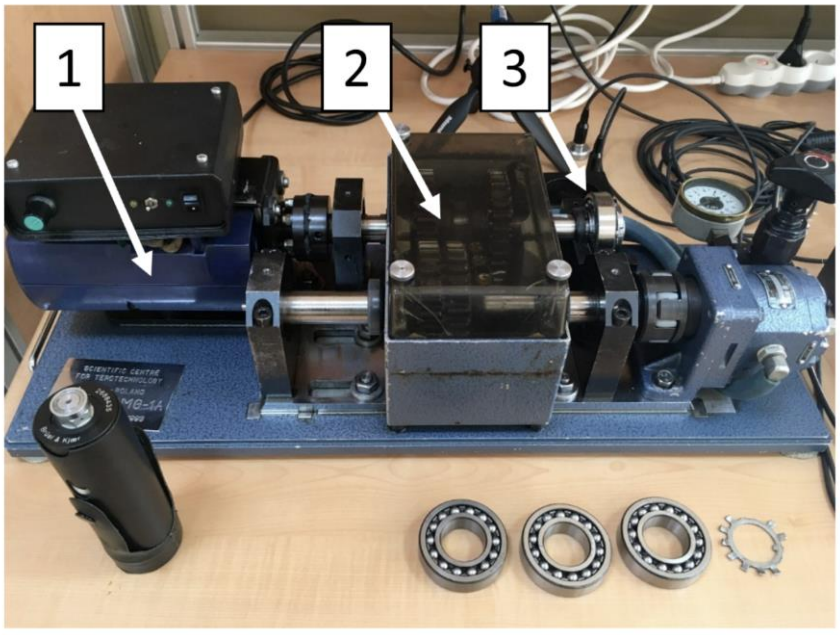

Figure 1 - Laboratory bench

Acceleration signals were captured from the bearings support to analyze the behavior of wheel bearings under operational stress. These signals are important for diagnosing the health of bearings and predicting potential failures. The measurement system collected data directly into MATLAB solution[2].

The signal nature of well-functioning and defective wheel bearings was closely examined, as visualized in the attached figures, where Fig. 2 represents a well-working wheel bearing signal, and Fig. 3 shows a signal from a defective wheel bearing. The differences in signal characteristics are crucial for understanding the condition of the bearings.

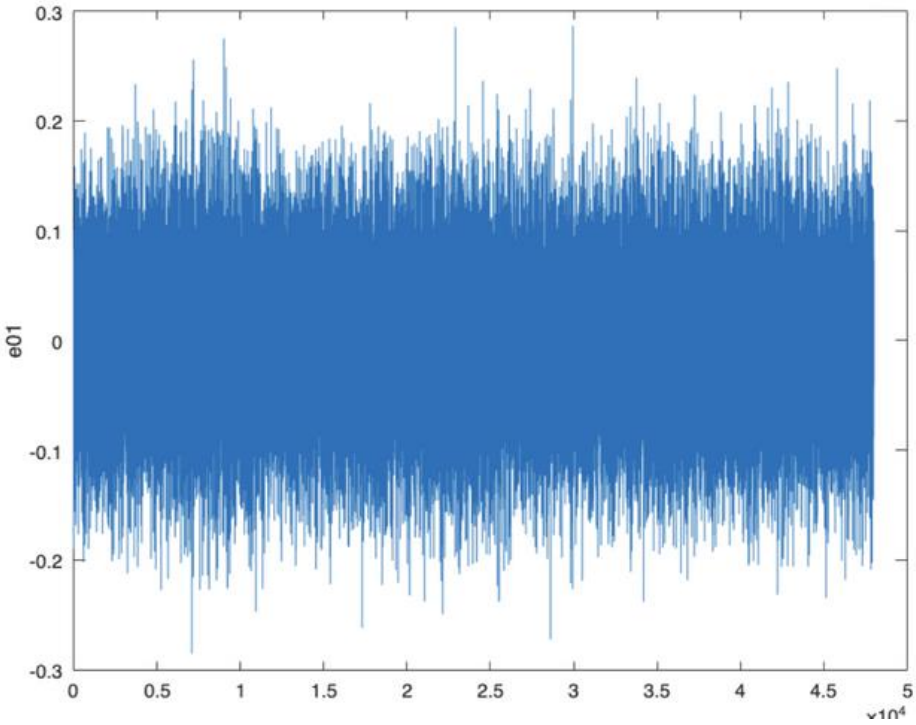

Figure 2 – Processed signal from the well-working wheel bearing

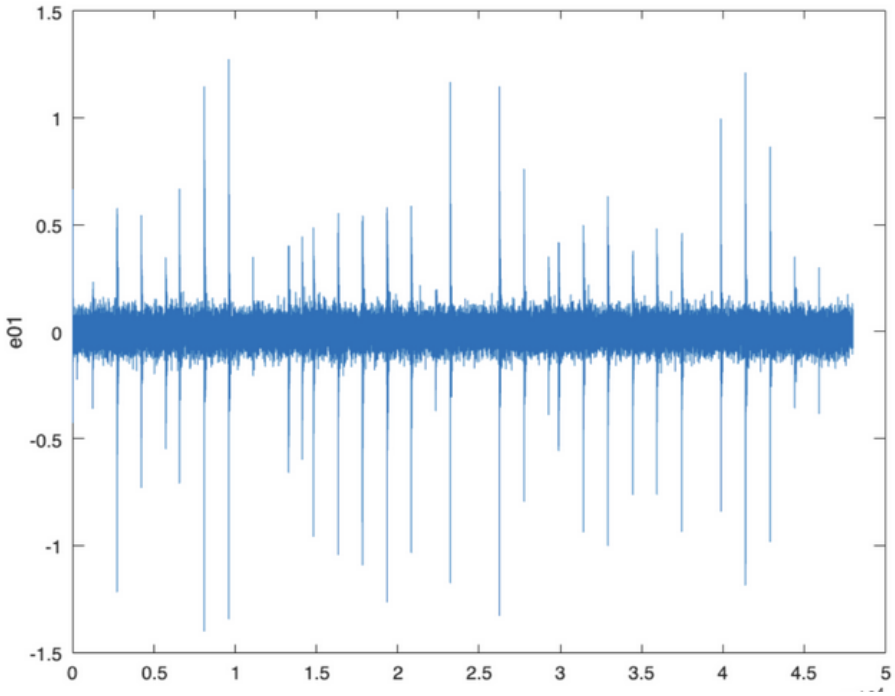

Figure 3 – Processed signal from the damaged wheel bearing

2.2 Data Analysis and Visualization

Before features can be extracted from the signal, an additional technique that called windowing should be applied to the signal. Windowing is a process where the signal is segmented into smaller parts, this segmentation allows for detailed inspection of the signal's attributes within manageable data portions. The choice of window size and overlap parameters often requires careful tuning to optimize performance and achieve more accurate results.

The results of the initial signal processing procedure are visualized in Figures 3 and 4, showcasing the Power Spectral Density (PSD) plots for both well-working and defected bearings. The PSD allows us to assess how the power of a signal is distributed across different frequency components, and can be used to analyze the frequency characteristics of time-domain signals, such as vibration signals from bearings. In formal terms, the PSD is derived from the Fourier Transform and is represented as the squared magnitude of the signal's Fourier Transform, normalized by its bandwidth [Link to literature].

Given a time-domain signal $x(t)$, the Power Spectral Density (PSD) - $S_x(f)$, is defined as:

$$S_x(f) = \lim_{\tau \to \infty} \frac{1}{\tau} E\left[\left|\int_{-\tau/2}^{\tau/2} x(t) e^{-j2\pi f t} dt\right|^2\right]$$

Where:

f - represents the frequency;

τ - is the duration of the observation window;

E - denotes the expectation operator, in practice often approximated by ensemble averaging.

The difference between the well-functioning and defected bearings is not only reflected in the numerical data but can also be easily detected visually on the spectrograms. The calculated PSDs for each segment can be arranged sequentially along the time axis to form a spectrogram that visualizes how the frequency content evolves over time. The PSD values are displayed as an intensity plot, with time on the horizontal axis, frequency on the vertical axis, and power represented by the color intensity.

By analyzing spectrograms in Figures 3 and 4, the frequency domain provides clear distinctions between healthy and faulty bearings.

For well-working bearings, a uniform spectral distribution, with energy concentrated in specific harmonic frequencies, can be seen. That usually corresponds to normal operation. On the other hand, for defected bearings, some irregularities in the PSD can be traced on the spectrogram, such as higher energy content at unexpected frequencies. These irregularities are caused by impulsive forces due to defects, which produce a broader range of frequencies.

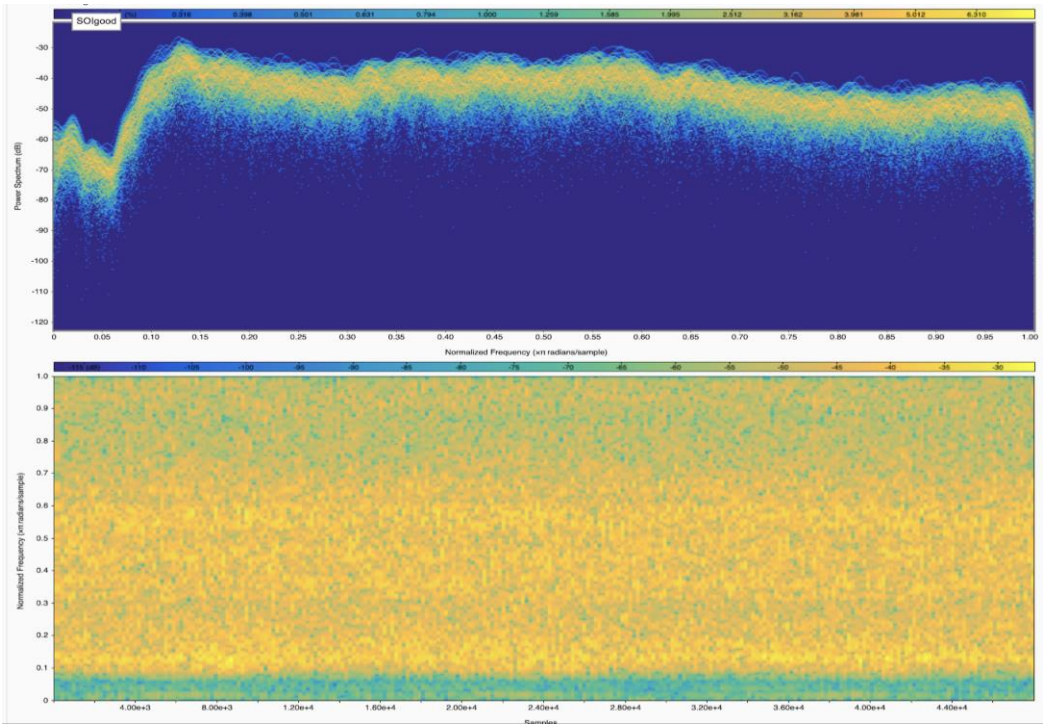

Figure 3 – Spectrogram for the good signal of interest

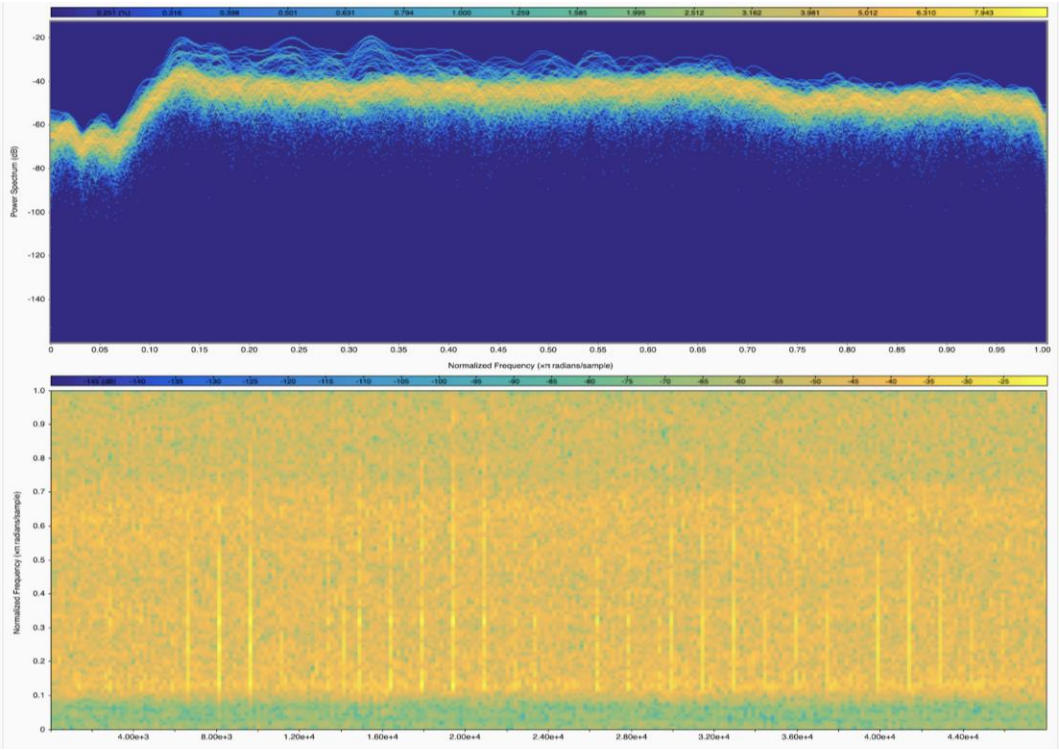

Figure 4 – Spectrogram for the faulted signal of interest

Before advancing to more sophisticated feature extraction methods and implementing machine learning techniques, it was decided to process the data using simpler statistical approaches to assess the baseline K-Means algorithm's performance. The windowed data underwent median and mean absolute deviation (MAD) calculations to obtain preliminary insights into the data's behavior.

For the subset of data designated for training, median values and mean deviations were computed to gain preliminary insights into the degree of variation within the windowed segments of the signal data. This initial analysis laid the groundwork for understanding the signal characteristics. The algorithm's predictive performance was tested by a subset of the data assigned specifically for testing.

Figure 5 illustrates the K-Means results using the median values, while Figure 6 displays the K-Means outcomes when employing mean deviations.

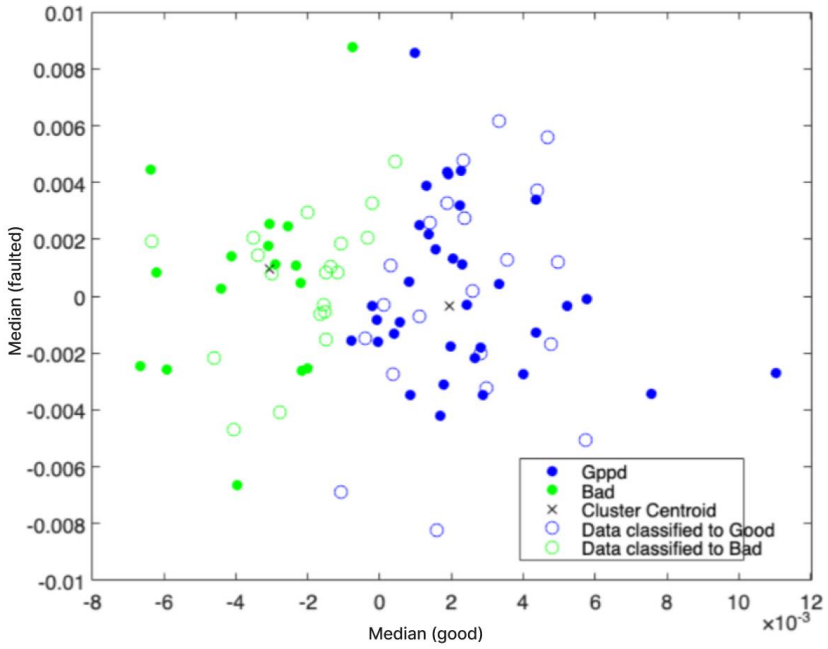

Figure 5 – K-Means based on median values

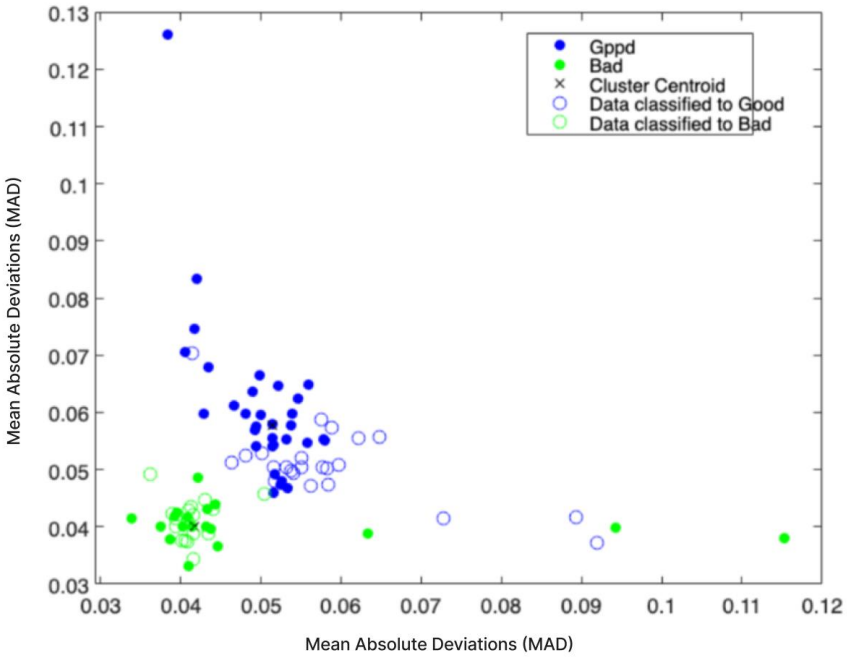

Figure 6 – K-Means based on the mean absolute deviation

During the analysis of the initial results, it was observed that relying solely on mean values, without accounting for mean deviations, led to fewer outliers being detected. This indicated that while mean values are effective for measuring central tendency, incorporating variability measures such as the Mean Absolute Deviation (MAD) could improve anomaly detection in signal data.

Based on these insights, the decision was made to use peak frequency and mean frequency, weighted by power, as the primary features for further SOM training. These features were chosen for their known ability to effectively capture the key characteristics of wheel bearing signals [11].

The peak frequency is the frequency at which the PSD has its maximum value for a given window.

$$f_{peak} = arg\ max_f\ S_x(f)$$

Where:

$S_x(f)$ - is the PSD as a function of frequency f.

The average frequency is the power-weighted mean of the frequencies in the PSD

for a given window. It is calculated as:

$$f_{mean} = \frac{\Sigma_i f_i S_x(f_i)}{\Sigma_i S_x(f_i)}$$

Where:

f_i - is the i-th frequency bin;

$S_x(f_i)$ - is the PSD as a function of frequency f.

These frequencies are fundamental characteristics in the signal analysis and can be used to distinguish between healthy and faulty bearings. The extracted features from both healthy and faulty bearings provided a comparative basis for fault diagnosis.

2.2 Algorithm Implementation and Outcomes Comparison

Based on the approach described above, utilizing MATLAB's signal processing toolkit, the algorithm presented in figure 7 was implemented.

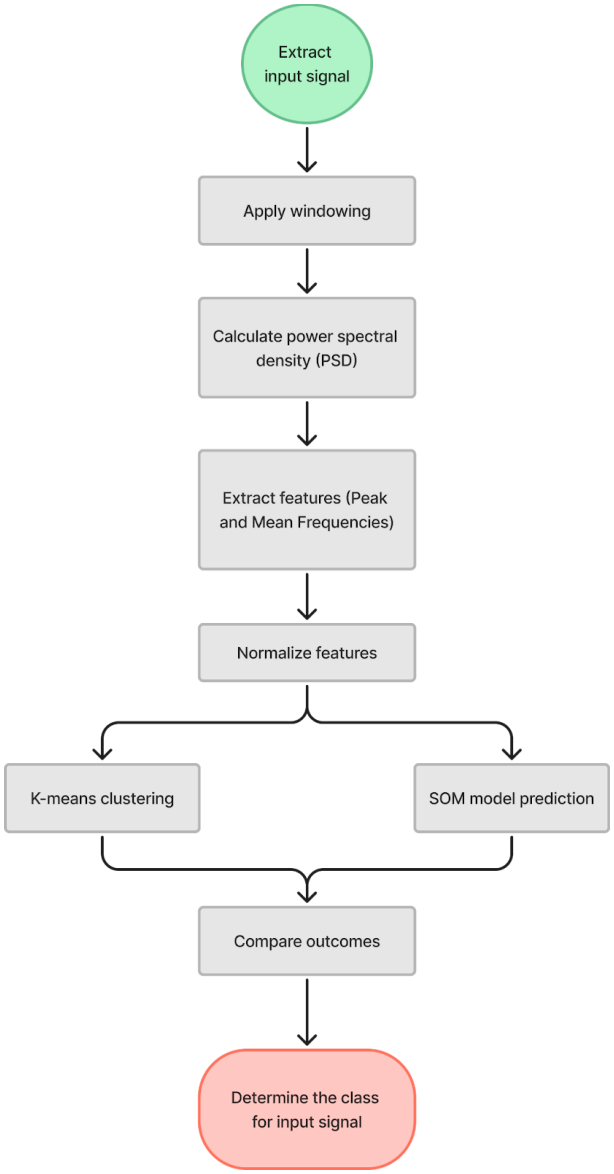

Figure 7 – Algorithm diagram

The process begins with extracting the input signal and capturing raw data from the experimental setup. The signal is then windowed, a window size of 1024 samples and an overlap of 128 samples were determined as the most efficient parameters, based on the practical experiment.

The feature extraction algorithm was implemented using MATLAB language, applying the Fast Fourier Transform (FFT) to each windowed segment to obtain frequency domain data. This transformation is followed by the calculation of the power spectral density (PSD) obtained using the Welch method, which aids in identifying significant frequency components for the algorithm[5].

At the next stage, the feature extraction was conducted on these windowed segments, where each window's peak frequency and average frequency were calculated.

Afterward, the normalization was applied to extracted features, as it is vital to maintain the consistency of the input to the machine learning algorithms.

The SOM was trained using MATLAB's neural network toolbox, where a 8x8 grid was chosen for mapping the input space. The training data was then fed into the network, this training process aimed to create a model that could effectively categorize the condition of the bearings based on the extracted features.

For testing the SOM model's performance, a separate dataset was used. The testing outcomes were benchmarked against the K-Means results, with the following mean results observed for K-Means:

Metric	**Value**
Precision	0.63158
Accuracy	0.63636
Recall	0.92308
F1 Score	0.75

Table 1 – K-means prediction results

In figure 8, the K-Means clustering results are shown, and in figure 9 the confusion matrix from testing data can be seen.

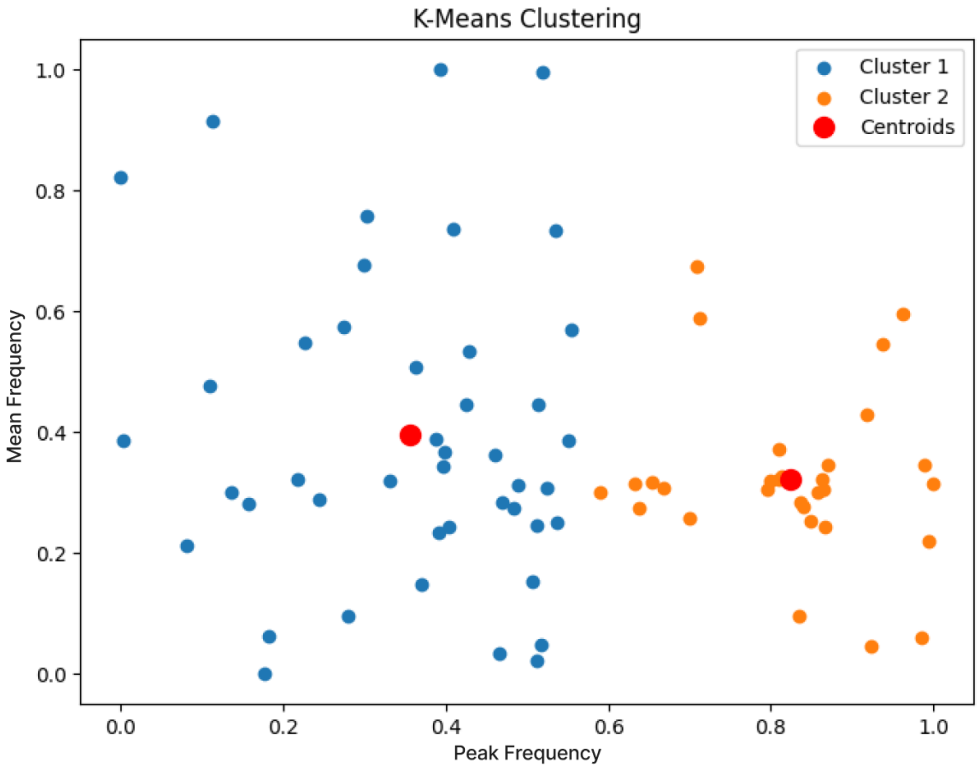

Figure 8 – K-Means clustering results utilizing mean and peak frequency

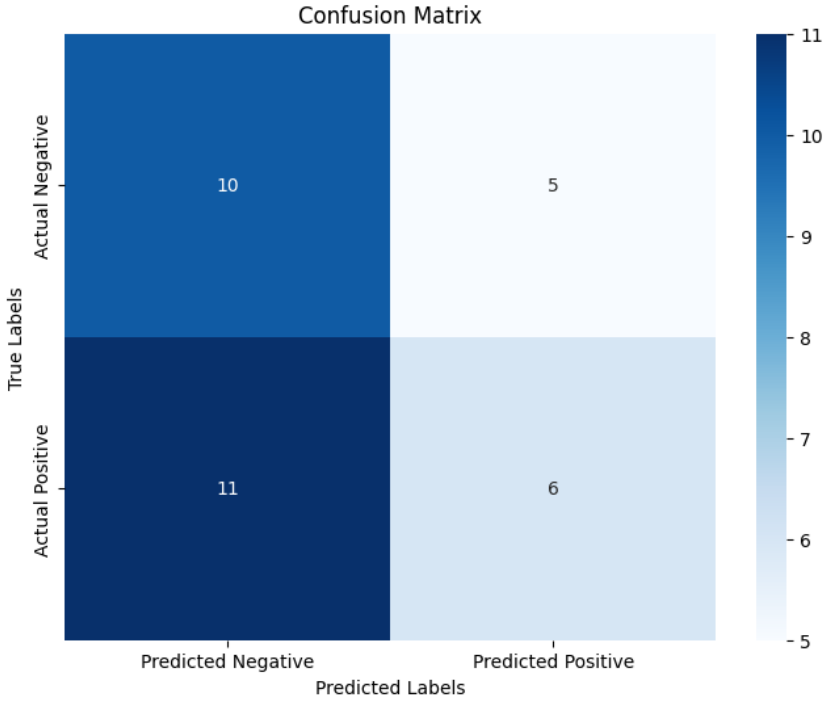

Figure 9 – Confusion matrix for K-means clustering

In figure 10 and 11, the visualizations of the SOM, such as SOM weights positions, and weights planes are presented respectively. SOM weights positions plot shows the arrangement of weights within the SOM network, highlighting the topology and how the map clusters the input data [6]. Blue dots represent neurons or nodes in the SOM that correspond to centroids of clusters. Red lines show neighboring relationships between neurons in the SOM, preserving the topology of the input feature space. Axes indicate the two-dimensional positions of neurons in the feature space, where each weight corresponds to an input feature's contribution to the clustering process.

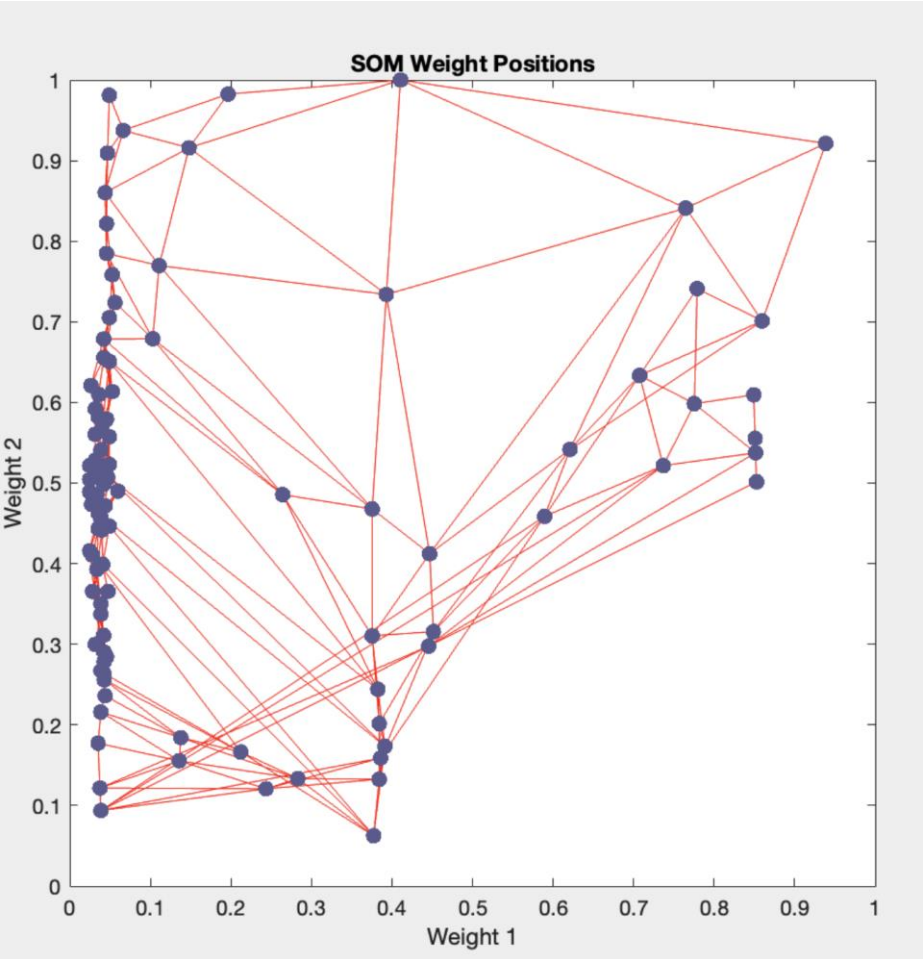

Figure 10 – SOM Weights Positions

The weights plane plot in figure 11 visualizes the connections between inputs and the SOM layer's neurons, with the color scale ranging from light to dark, indicating

the distance from neighboring neurons, also known as the Unified Distance Matrix. Darker regions suggest higher distances between neurons, representing well-separated clusters in the feature space, while lighter regions indicate more similarity between neighboring neurons. These dark regions highlight the SOM's ability to identify distinct groups of "good" and "bad" data points, whereas overlapping markers in lighter regions suggest areas with less confident classification.

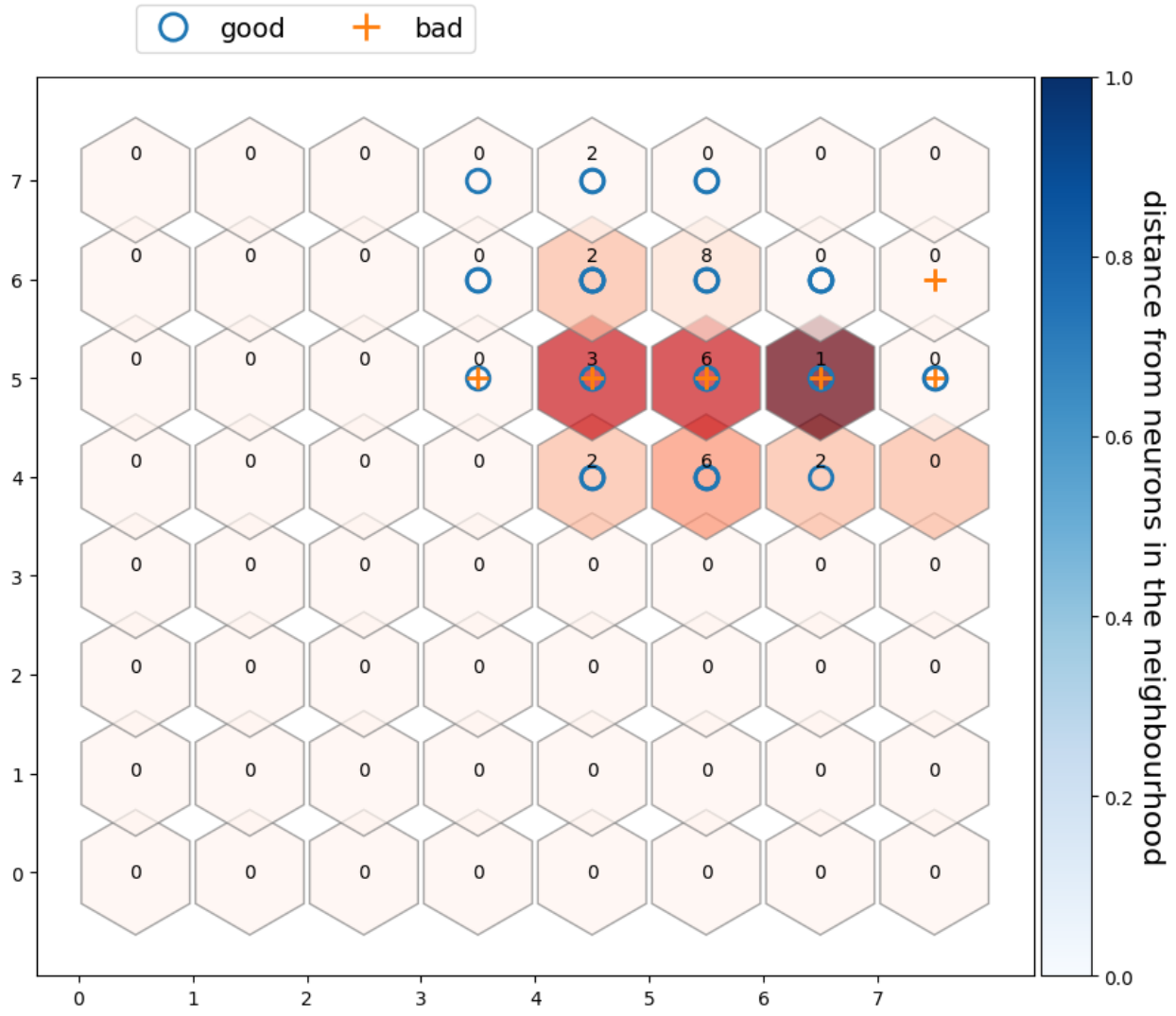

Figure 11 – SOM Weights Planes

The results obtained from the SOM testing demonstrated a notable improvement over K-Means clustering results:

Metric	Value
Overall Accuracy	0.95
Macro Avg Precision	0.96
Macro Avg Recall	0.93
Macro Avg F1-score	0.94
Weighted Avg	0.95

Table 2 – SOM prediction results

The SOM's performance metrics, particularly in precision and recall, highlight its efficacy in the automatic diagnosis of wheel bearing faults. These results affirm the advantages of using advanced machine learning techniques, such as SOM, for improving fault detection accuracy in mechanical systems.

3. SOFTWARE STRUCTURE

The software structure of the project is guided by the principles outlined in "Fundamentals of Software Architecture: An Engineering Approach" by M. Richards and N. Ford [9], and the "Guide to the Software Engineering Body of Knowledge (SWEBOK)" by Bourque and Fairley [7]. This section provides an elaborate examination of the system's architecture, the user interface and interaction flow, and the strategies for system deployment.

Additionally, the implementation methodology and development process of this Capstone Project was structured to transition smoothly from a successful

prototyping phase to a robust production phase, adhering to agile development practices. In the first phase, the project achieved a functional prototype in MATLAB. This prototype served as proof of concept, providing a practical demonstration of the algorithm's ability to detect wheel bearing faults. With the completion of the prototyping phase, the focus has shifted towards developing the software for production usage. This transition involves the reimplementation of MATLAB prototype scripts and SOM model into production-ready code using RUST and Python, chosen for their performance and adaptability.

3.1 System Architecture

For a software system dedicated to the detection of wheel bearing faults, focusing on specific software qualities is crucial to ensure its effectiveness, performance, and usability. Based on the nature and requirements of such a system, the following three software qualities are particularly important:

Software Quality	Description
Performance	High processing speed for real-time data analysis and fault detection. Efficient resource utilization to manage large datasets and complex calculations without significant delays or system overloads.
Scalability	Ability to adapt to increasing data volumes and more complex analysis scenarios without compromising performance. Flexibility to integrate with various hardware and software environments, allowing for expansion or modification as needed.

Usability	The ease with which users can interact with the software to efficiently and accurately diagnose wheel bearing faults. This encompasses an intuitive user interface, clear data visualization, and straightforward navigation.

Table 3 – Software Qualities

In addition to selected software qualities, the following assumptions were considered when making architecture significant decisions:

1. The input data, primarily acceleration signals from wheel bearings, will be of high quality and sufficiently detailed for analysis.
2. Users of the system have a basic understanding of mechanical diagnostics, enabling them to interpret the data visualizations and diagnostic results accurately.
3. The operating environment for the software will have the necessary computational resources to support data processing and machine learning tasks.
4. Regular maintenance and updates of the system is necessary to ensure that the software remains up-to-date with the latest advancements in technology and methodologies.
5. Integration with existing diagnostic tools and systems should be possible, allowing for a more comprehensive approach to machinery maintenance.

The selected architecture style for the system is – Monolith architecture combined with pipes and filters elements for data processing.

The system's architecture is designed to address the complexities of modern web applications, the system includes a RESTful API for efficient client-server communication and modular components for scalability and maintainability, utilizing a combination of high-performance programming languages including Python and Rust, and different front-end technology. This combination ensures that the system is not only efficient in processing and analyzing data but also user-friendly and accessible.

The initial draft of the system architecture depicted in figure 12 presents a foundational blueprint for the wheel bearing defect detection application. This draft outlines a user-centric design where the user uploads data to the front-end interface, built with robust web technologies.

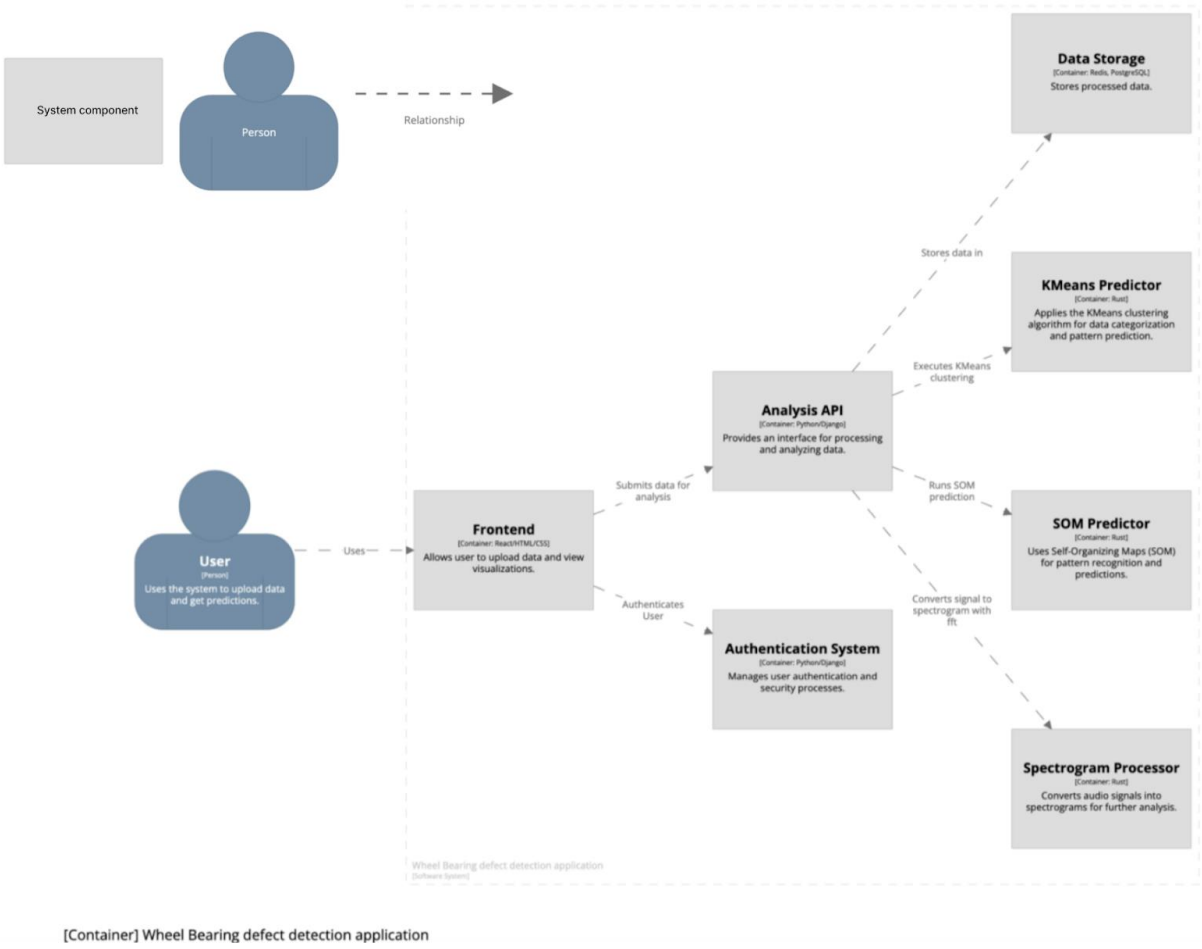

Figure 12 – Initial C4 model diagram of the system architecture

The final architecture, illustrated in figure 13 as a model diagram, offers a comprehensive view of the wheel bearing defect detection application's structure. This model provides a detailed representation of the system. It includes user interactions with the front-end, where they can upload data and receive predictions. The front-end connects to various back-end services, including an Analysis API that processes the data, an Authentication System that secures user interactions,

and separate components for back-end development, the project leverages Python and Rust for their robustness in handling large-scale data processing and machine learning tasks. The system architecture facilitates seamless data flow and integration between the front-end and back-end components SOM predictions and visualization that handle the core analytical processing. There is also a dedicated Spectrogram Processor to transform signals into visual formats conducive to further analysis.

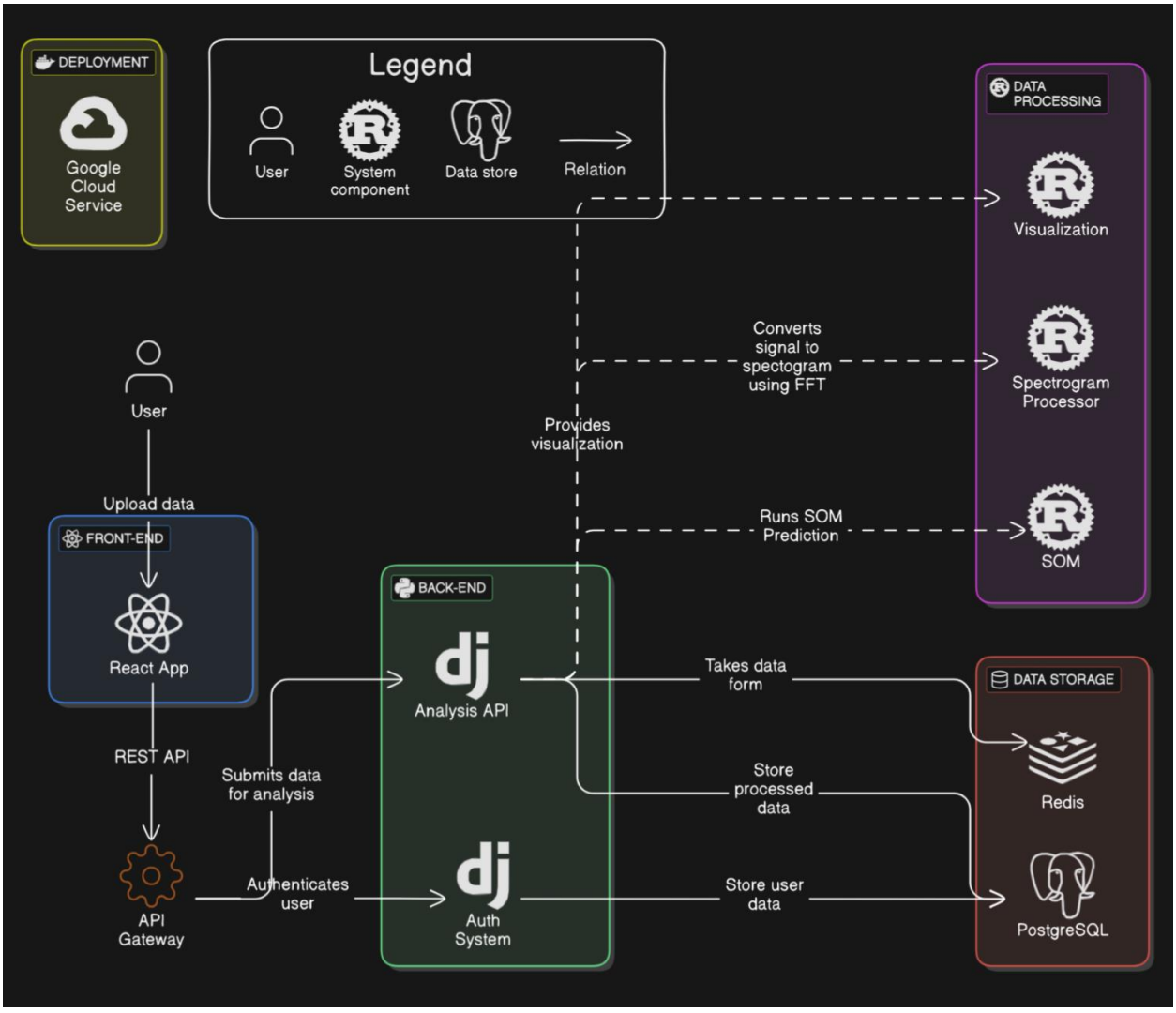

Figure 13 – Final model of the system architecture

In addition to an advanced approach with back-end and front-end implementation, the system architecture facilitates seamless data flow and integration between the front-end and other system components by utilizing the following components: background scheduler, Dramatiq, Redis, and PostgreSQLv16.

Dramatiq is a distributed task processing library that works well with Django and enhances the application's ability to handle background processes and batch jobs efficiently. Also, this service natively integrates Rust-based modules, that were created to predict results with SOM. Redis is employed as a cache system and a message broker, Redis provides high-speed data caching to improve the system's performance by reducing the time needed to access frequently requested data. It also facilitates message brokering for Dramatiq, contributing to the effective management of task queues. PostgreSQL, a powerful, open-source object-relational database system known for its reliability, feature robustness, and performance. It serves as the primary data storage solution, housing all the application data including user information, analysis records, and system logs.

The selection of the Django framework for the backend of the wheel bearing defect detection application was a strategic decision underpinned by Django's reputation for enabling rapid development with a clean and pragmatic design. It provides a secure way to manage user interactions and reliably process data, which is essential for the system's API service. Django's compatibility with various databases and its built-in ORM (Object-Relational Mapping) facilitate efficient data operations and manipulation, particularly with PostgreSQL, which is used in this project for data storage. Moreover, Django's scalability and flexibility allow the backend to accommodate future enhancements and integrations, ensuring that the system can evolve in line with emerging requirements and technologies.

To further enhance the performance of the application, particularly in the computationally intensive tasks of signal processing and machine learning, a Rust library was created using PyO3. PyO3 is a Rust library that facilitates the creation of Python modules in Rust, providing the ability to write Python extensions that execute at high speed. By leveraging Rust's performance, the application benefits from the efficiency and speed of Rust's compilation and execution.

The use of Rust's high-performance libraries, such as rusticsom for Self-Organizing Maps and rustfft for Fast Fourier Transforms, guarantees that the application can handle the processing of large datasets and complex algorithms

with ease. The approach of integrating of these Rust libraries with Python through PyO3, optimizes the system's performance and leverages the strengths of both languages—Python's simplicity and extensive libraries for web development, and Rust's speed and memory safety for performance-critical tasks.

It was decided to use a REST API architecture for the solution to ensure scalability, flexibility, and maintainability. The REST design, defined by constraints such as resource identification, manipulation through representations, self-descriptive messages, and hypermedia as the engine of application state (HATEOAS) [13], aligns well with the system's requirements. Combining Django and the HTMX library, the application fully implements HATEOAS principles, keeping the client decoupled from the server. This decoupling enables server functionality to evolve independently while maintaining compatibility [14]. The REST API dynamically provides hypermedia links, allowing efficient resource navigation without relying on hard-coded endpoints. By combining Django's robust backend capabilities with HTMX's dynamic frontend interactions, the system offers a seamless user experience, enhanced scalability, and adaptability to evolving requirements.

3.2 Interface and Interaction Flow

The design and interaction flow of the wheel bearing defect detection application are built with user experience at the core, drawing on best practices from Barnum's "Usability Testing Essentials" [10].

Upon entering the system, users are welcomed with a dashboard that acts as the central hub for all key functionalities. This interface simplifies navigation, offering immediate access to initiate new analyses or review past records. As illustrated in the initial UX wireframe in figure 14, the dashboard is meant to be both intuitive and informative. It provides users with a clear overview of the system's functionality while emphasizing ease of use, enabling seamless access to critical features such as starting new analyses or revisiting historical data.

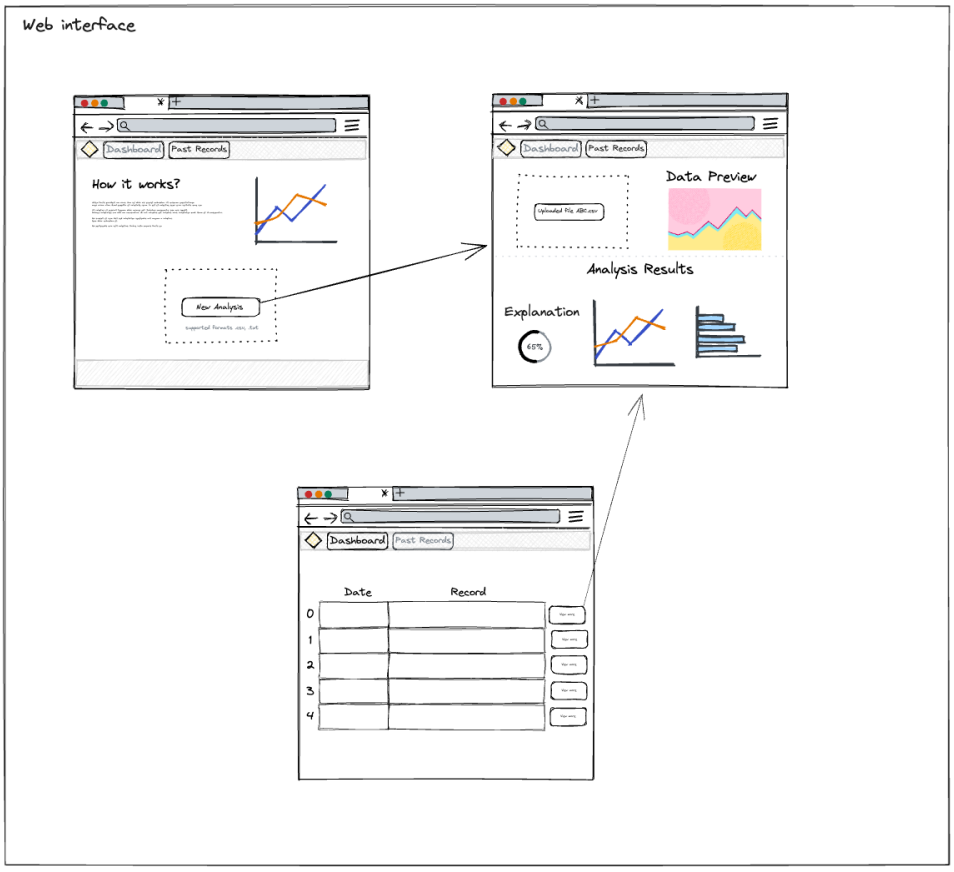

Figure 14 – Web interfaces UX

For the user interface and data visualization components, the project employs a combination of Python with the Django framework, HTMX, and React.js. Python and Django will provide a solid backend structure, facilitating data management and server-side processing. HTMX adds dynamic functionality to the web interface, enabling smooth and responsive interactions without requiring full-page reloads. React.js, a popular JavaScript library for building user interfaces, is utilized to create a responsive and intuitive frontend. This combination will enable users, such as mechanics and engineers, to interact with the system effortlessly, providing them with real-time diagnostic data and insights. In figure 15, the implemented interfaces are presented.

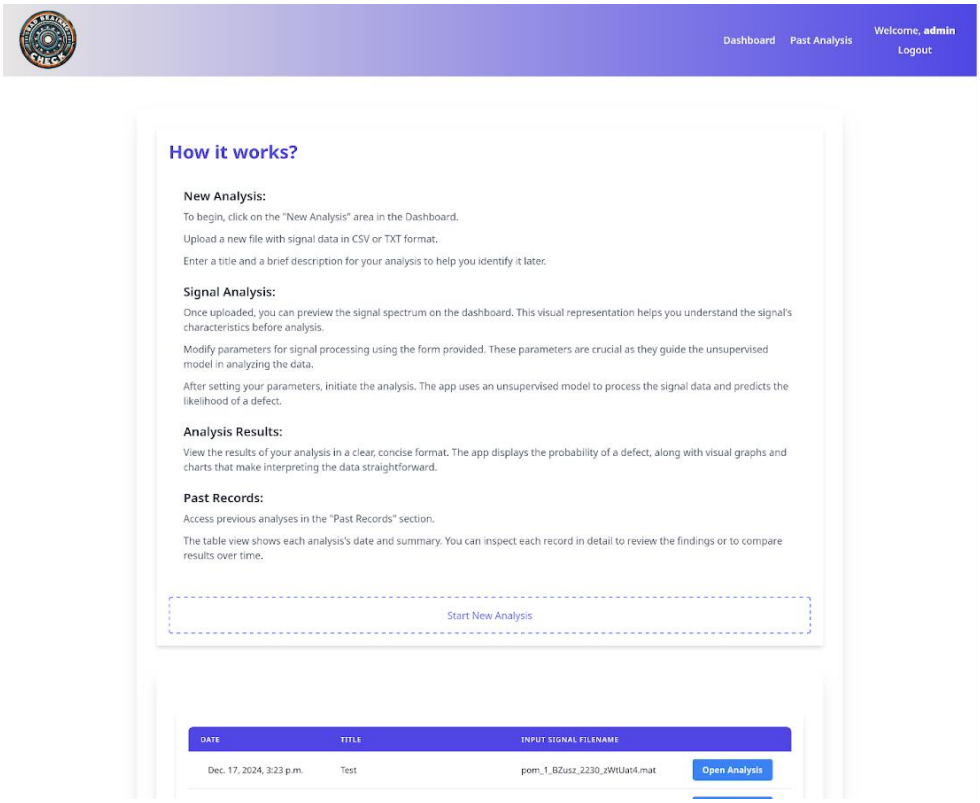

Figure 15 – Interface for a dashboard screen

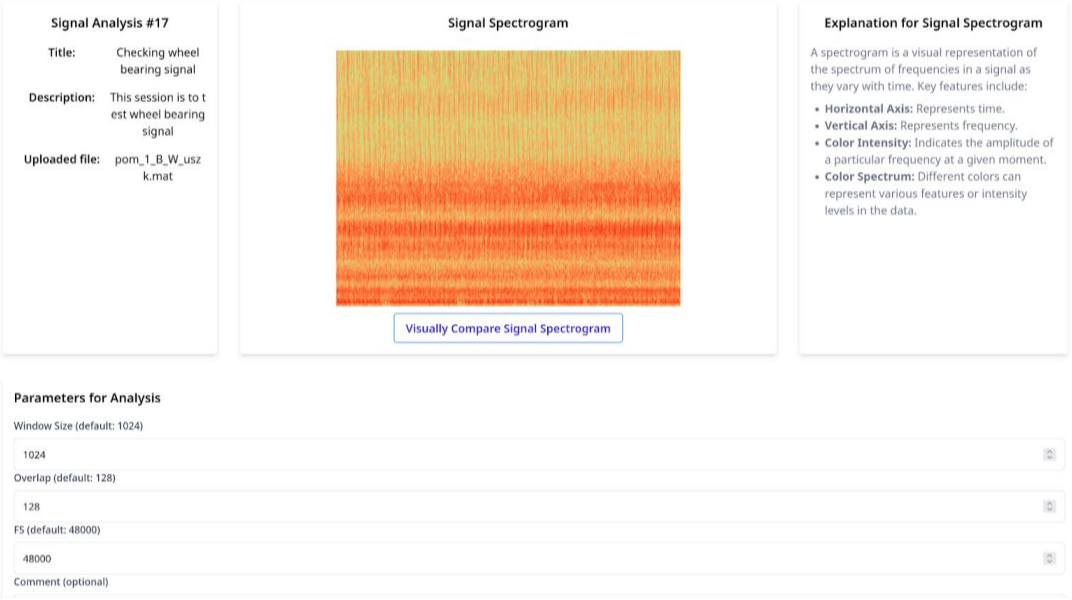

Figure 16 – New analysis screen

The user's journey through the application is designed to be logical, intuitive, and user-friendly, as illustrated in figure 17. After the initial welcome screen, users are presented with clear options to either sign up or sign in, seamlessly transitioning them to the main dashboard. From there, they can choose to explore past analyses or start a new one, with each step for uploading files and setting parameters thoughtfully structured to ensure clarity and ease of use. The signal analysis process prioritizes transparency and user engagement, culminating in a detailed and informative presentation of results. Each action is designed to make it easy for users to use.

Barnum's insights on usability testing emphasize the significance of this approach, ensuring that the interface accommodates users of varying technical backgrounds [10]. The user interaction flow guides individuals through a logical progression of actions, from uploading data to receiving analysis results, with every step providing the necessary information and controls. This UX design prioritizes simplicity and functionality, creating a smooth, efficient process that allows users to leverage the application's full potential with confidence and ease.

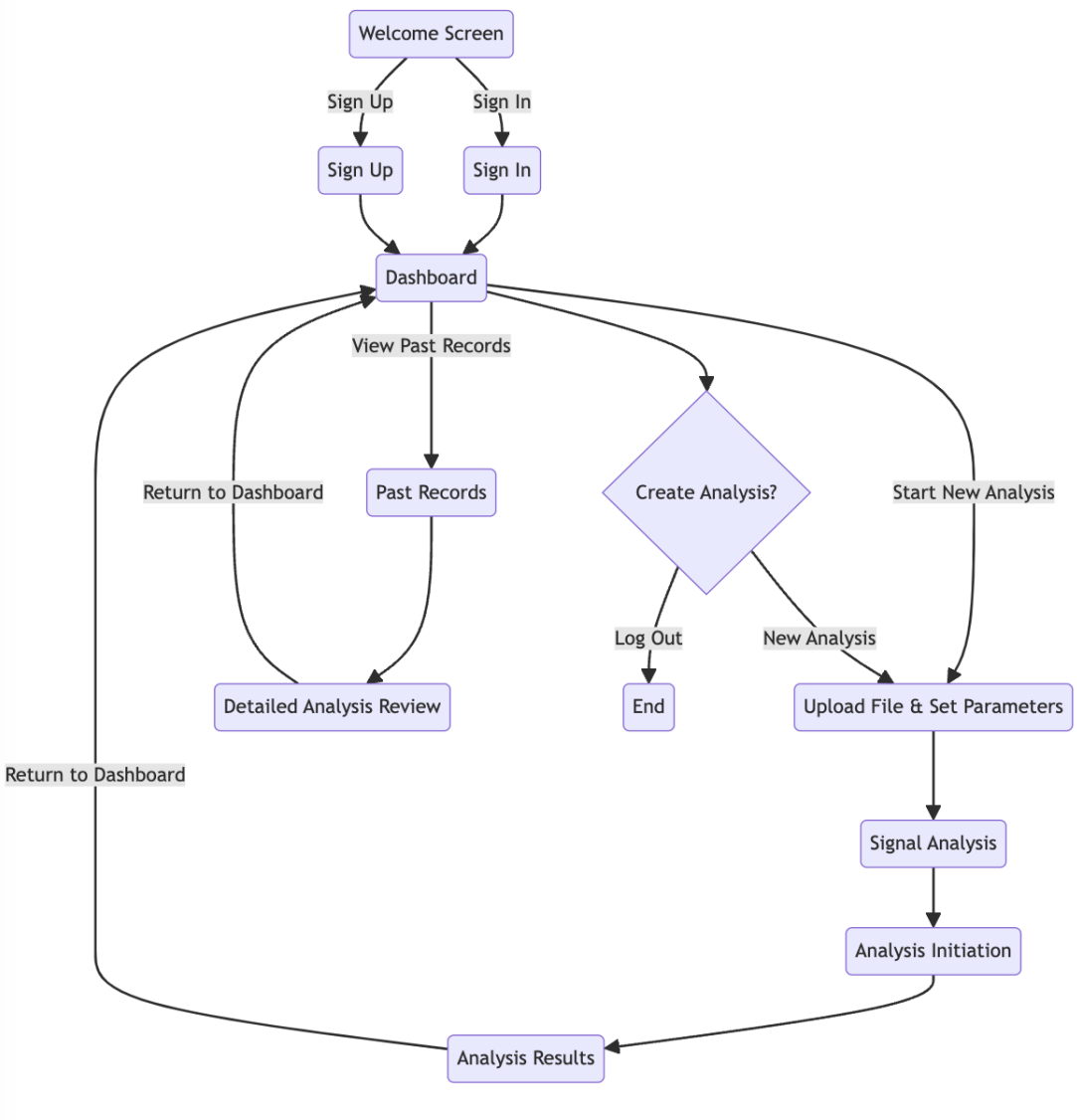

Figure 17 – User interaction flow

3.3 System Deployment

The deployment strategy for the application leverages the scalability and reliability of cloud computing, with Google Cloud as the chosen platform. The entire application stack is containerized using Docker, enabling consistent and isolated environments for each service, as presented in figure 18. This containerization ensures portability, simplifies updates, and facilitates consistent testing while

allowing flexible scalability to meet varying user demands. By deploying on a single Google Cloud instance, the system effectively utilizes cloud resources to deliver a flexible and reliable solution suited to the wheel bearing defect detection application.

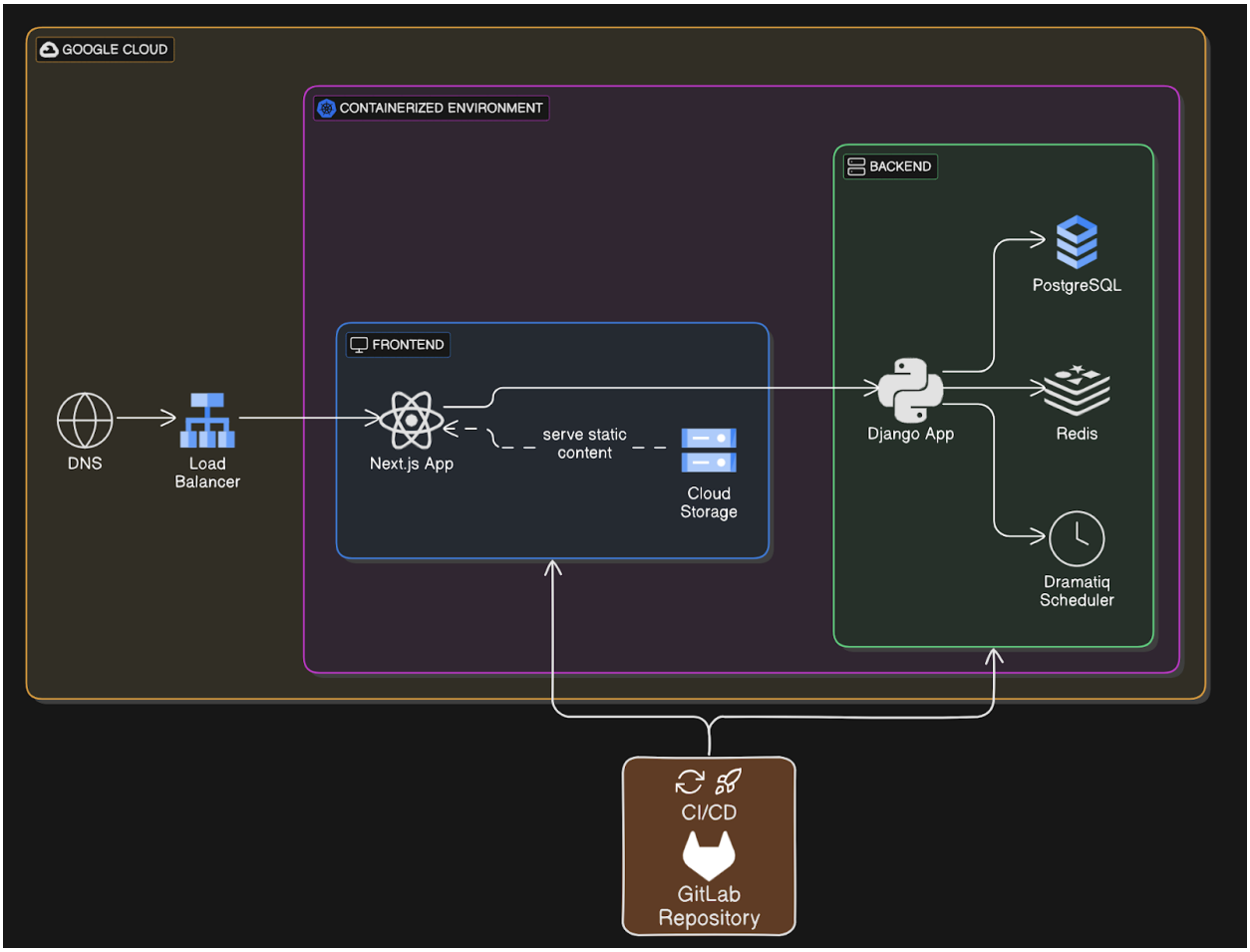

Figure 18 – System deployment diagram

To enhance the development lifecycle, GitLab is employed as the central DevOps tool, providing a comprehensive solution for version control, collaboration, continuous integration, and deployment [7]. This modern approach combines the robustness of Docker, the scalability of Google Cloud, and the efficiency of GitLab to create a straightforward, scalable, and maintainable deployment process, ensuring a reliable service delivery.

4. RESULTS

The primary goal of this Capstone Project — to develop an innovative software solution for early detection of wheel bearing defects in rotating machinery — has been successfully achieved. By combining advanced signal processing techniques with machine learning algorithms, the project delivered a comprehensive, scalable, and user-friendly application for automated fault diagnosis. This section reviews the extent to which the project objectives were met and highlights key findings.

Foremost, the project effectively addressed the key questions identified during its initial planning. Peak and mean frequency emerged as the most reliable features for distinguishing between healthy and defective bearings, providing a strong foundation for accurate classification. Although K-means was useful for initial categorization, it demonstrated limitations in dealing with complex fault patterns and overlapping clusters. In contrast, SOMs not only delivered superior accuracy but also offered greater interpretability, establishing their effectiveness for advanced fault detection.

Based on the developed algorithm, an ML model based on the Self-Organizing Maps (SOMs) was trained. The model showed superior metrics on the dataset such as an overall accuracy of 95%, a macro-average precision of 96%, and a recall of 93%. These results further illustrate the effectiveness of SOMs in handling non-linear relationships within the data.

Secondly, the project's architecture successfully integrated the signal processing and machine learning modules into a modular monolith structure. Built using Python/Django and Rust for the backend and HTMX with React.js for dynamic interactions, the application conformed to REST API principles, ensuring scalability and performance. The deployment pipeline, established on Google Cloud and powered by GitLab, provided a solid environment for the application, enabling seamless updates and scaling to handle varying loads.

Finally, the goal was to make the system easy to use for engineers and maintenance workers. This was done using well-designed user interfaces that can be seen in the following figures.

The application includes unique features that can't be found in existing solutions, including the Spectrogram Comparison Interface depicted in figure 19. This feature allows users to visually assess the uploaded signal against known references, making it easier to detect patterns, anomalies, or faults in the data and discover additional diagnostic insight.

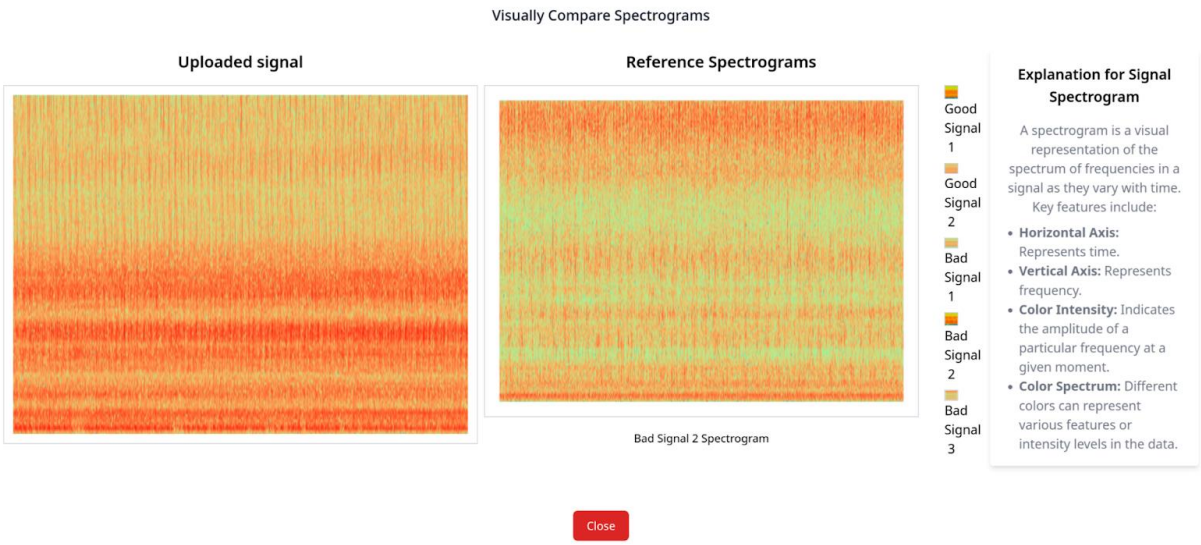

Figure 19 – System deployment diagram

Also, the application provides a dedicated interface for adjusting analysis parameters such as window size, overlap, and sampling frequency (FS) to fine tune the signal processing to specific requirements.

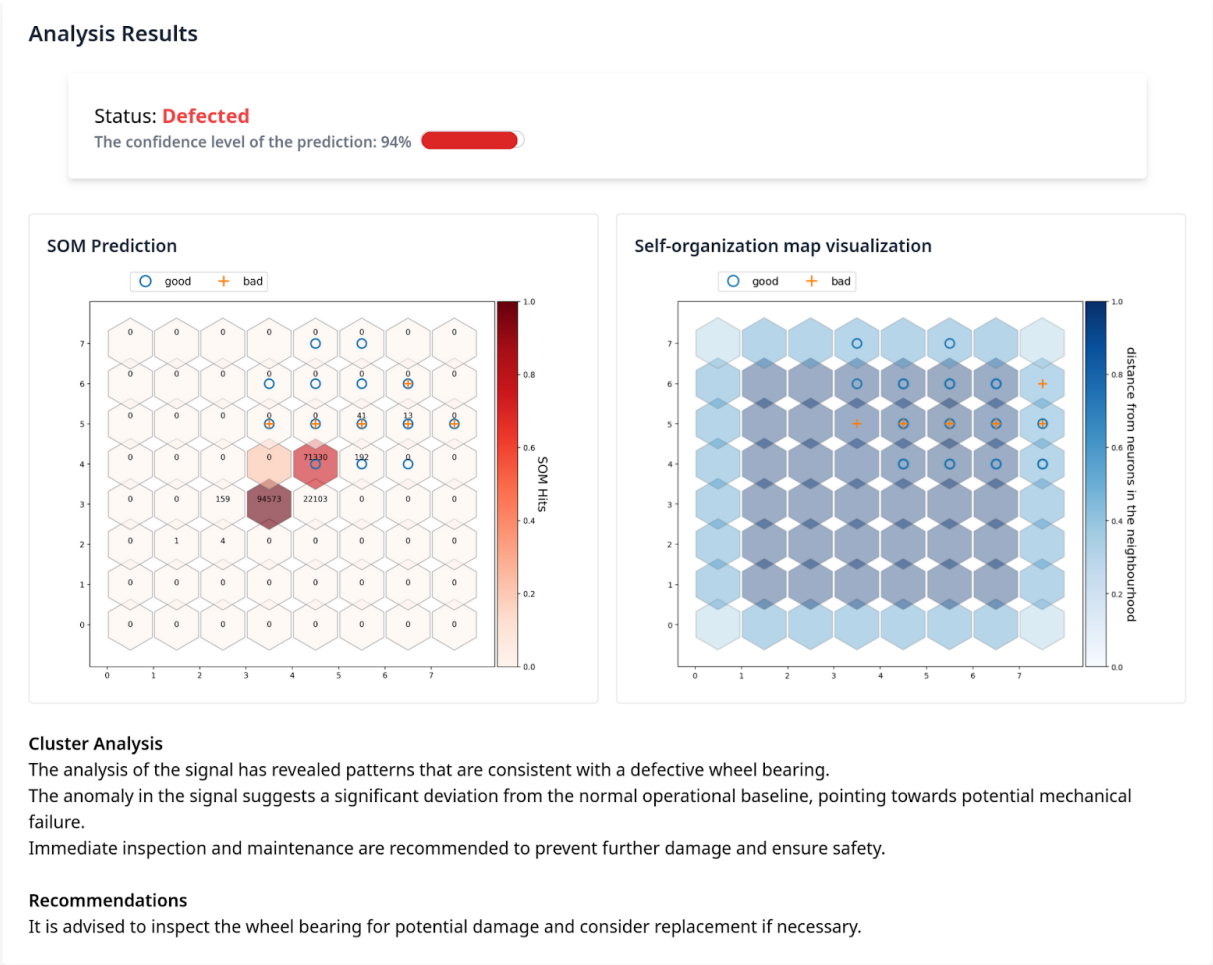

Figure 20 – The analysis results interface

The Analysis Results Interface, presented in figure 20, serves as the final step in the diagnostic process, providing a full overview of the system's findings. At the top, the status of the analysis is prominently displayed, indicating whether the wheel bearing is "Defected" or "Healthy," along with the confidence level of the prediction, ensuring clear and immediate feedback for users. Below, two detailed visualizations—SOM Prediction and Self-Organizing Map (SOM) Visualization—offer more in-depth insights into the signal's classification. The SOM Prediction diagram highlights clusters of "good" and "bad" signals, helping users interpret the diagnostic patterns, while the U-Matrix visualization shows the distances between neurons, reflecting the clustering quality.

This interface also includes a Cluster Analysis section that explains the results in plain language, and a Recommendations section that provides actionable advice,

such as inspecting or replacing the wheel bearing. Together, these elements enhance the application's usability and effectiveness, making it a practical tool for real-world diagnostics.

5. CONCLUSION

In summary, this project brings together state-of-the-art technologies in software engineering and signal processing, setting a new standard in the field of machine maintenance and fault diagnosis. This technological innovation can revolutionize how industries approach machinery maintenance, making it a significant step forward in academic research and practical application.

The developed application fulfills all the major objectives of the project. It provides a scalable, maintainable, and user-friendly platform for wheel bearing fault detection. The excellent performance of SOMs, combined with the intuitive design of the user interfaces, makes the application not only effective but also accessible to its target users. By utilizing technologies as SOM, Django, React.js, HTMX, PostgreSQL, and Rust libraries for enhanced performance, this Capstone Project demonstrates a practical and innovative approach to early fault detection in rotating machinery.

6. REFERENCES

[1] Duda, Jarek & Leśkow, Jacek & Pawlik, Paweł & Cioch, Witold. (2024). CMAFI — Copula-based Multifeature Autocorrelation Fault Identification of rolling bearing. Mechanical Systems and Signal Processing. 211. 111221. 10.1016/j.ymssp.2024.111221.

[2] Cioch, W., Knapik, O. and Leskow, J. (2013), Finding a frequency signature for a cyclostationary signal with applications to wheel bearing diagnostics, Mechanical Systems and Signal Processing, vol 38, pp. 55 - 64.

[3] Smith, W.A., Borghesani, P., Randall, R.B., Antoni, J., El Badaoui, M., Peng, Z. (2024). High-speed bearing diagnostics: Observations from the Surveillance 8 Safran contest data. Mechanical Systems and Signal Processing, 216, 111484. https://doi.org/10.1016/j.ymssp.2024.111484.

[4] Tandon, N., and Choudhury, A. (1999). A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. Tribology International, 32(8), 469-480.

[5] Wodecki, J., Michalak, A., and Zimroz, R. (2018), Optimal filter design with progressive genetic algorithm for local damage detection in rolling bearings, Mechanical Systems and Signal Processing, vol. 102, pp. 102–116.

[6] J. Tian, M. H. Azarian, and M. Pecht, “Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm”, PHME_CONF, vol. 2, no. 1, Jul. 2014.

[7] Bourque, P., and Fairley, R.E. (Eds.). (2014). Guide to the Software Engineering Body of Knowledge (SWEBOK), Version 3.0. Los Alamitos, CA: IEEE Computer Society.

[8] Wang, M., Chen, Y., Zhang, X. *et al.* (2022), Roller Bearing Fault Diagnosis Based on Integrated Fault Feature and SVM. *J. Vib. Eng. Technol.* **10**, 853–862.

[9] M. Richards and N. Ford, "Fundamentals of Software Architecture: An Engineering Approach," 1st ed., O'Reilly Media, Inc., Jan. 2020.

[10] Barnum, C. M. (2020). Usability Testing Essentials: Ready, Set ...Test!, 2nd Edition. Burlington, MA: Morgan Kaufmann Publishers.

[11] Jérôme Antoni, Cyclostationarity by examples, Mechanical Systems and Signal Processing, Volume 23, Issue 4, 2009, Pages 987-1036, ISSN 0888-3270, https://doi.org/10.1016/j.ymssp.2008.10.010.

[12] Randall, R.B., and Antoni, J. (2011). Rolling element bearing diagnostics—A tutorial. Mechanical Systems and Signal Processing,

[13] Fielding, Roy Thomas (2000). "Representational State Transfer (REST)". *Architectural Styles and the Design of Network-based Software Architectures* (PhD). University of California, Irvine. p. 82. ISBN 0599871180.

[14] HTMX.org. (2021). HATEOAS. Retrieved from https://htmx.org/essays/hateoas/