**Smart Guard Factory System:**

**Autonomous Maintenance through AI-Integrated IIoT**

**ITAI – 3377 – A.I. at the Edge & IIOT**

**Team A  
Professor: Patricia Mcmanus**

**08/08/2024**

**Capstone Project: Autonomous Agent and Generative AI for Edge and IIoT Systems**

**1. Project Proposal and Planning**

**1.Project Proposal**

The "SmartGuard Factory System" aims to revolutionize the way maintenance is conducted in smart factories by integrating advanced AI and IoT technologies. This project will develop an autonomous maintenance agent capable of real-time decision-making to optimize machine health and efficiency, reduce downtime, and extend the lifecycle of machinery.

**Objectives:**

* **Reduce Maintenance Response Times:** Utilize real-time data processing to detect and respond to potential machine failures instantly.
* **Increase Equipment Uptime:** Apply predictive maintenance strategies to foresee and rectify potential issues before they escalate into costly downtimes.
* **Improve Maintenance Efficiency:** Leverage AI to automate routine checks and maintenance tasks, freeing human resources for more complex problems.

**Innovation and Impact:**

* **AI-Driven Predictive Analytics:** Utilize vibration, temperature, and sound analytics to predict failures and maintenance needs.
* **Real-Time Alerting System:** Implement an SMS-based alerting mechanism for rapid human intervention when needed.
* **Enhanced Data Security:** Integrate robust cybersecurity measures to safeguard critical industrial data.

**2.Project Planning**

**System Components:**

* **Sensors:**
* Vibration: MakerHawk Analog Ceramic Piezo Vibration Sensors for detecting anomalies.
  + Temperature: DS18B20 Thermocouple and GY-906 MLX90614 IR Sensor for monitoring operational temperatures.
  + Sound: High Sensitivity Sound Detection Module for acoustic analysis of machinery.
* **IIoT Platform:** Utilize NVIDIA platforms (Nano, Orin, TX2) for edge computing capabilities.
* **Networking:**
  + Local: Wi-Fi and Ethernet for factory-floor connectivity.
  + Remote: SMS for urgent communications and alerts.
  + Power: Power Over Ethernet (POE) for reliable power supply and data transmission.

**2. System Design and Development**

**1. System Architecture Design**

1. **Data Flow Design:**

* **Sensor Integration:** Sensors on machinery collect real-time data on vibration, temperature, and sound. This data is crucial for detecting potential failures or maintenance needs.
* **Edge Processing:** Use NVIDIA platforms like Nano, Orin, or TX2 for initial data processing. This reduces latency and allows for quicker response times by processing data close to where it is generated.
* **Data Aggregation:** Data from multiple sensors and machines is aggregated at edge nodes before being sent to the central system for more complex processing and decision-making.
* **Communication Protocols:**
  + **Local Network:** Utilize Wi-Fi and Ethernet for internal data transfers within the factory.
  + **External Alerts:** Implement SMS-based alerts for urgent maintenance actions, ensuring quick human intervention when necessary.

**2. Security Measures:**

* **Encryption:** Implement strong encryption for data at rest and in transit to protect sensitive information.
* **Access Control:** Use robust authentication and authorization mechanisms to control access to the IoT devices, data streams, and edge computing resources.
* **Regular Updates and Patch Management:** Ensure that all system components, including operating systems and applications, are regularly updated to protect against known vulnerabilities.
* **Hardware:** Ensure all edge devices have anti-tamper seals.

**2. AI Model Development**

**1. Generative AI Techniques for Model Training:**

* **Synthetic Data Generation:** Use generative AI techniques to create synthetic datasets when real data is insufficient for training. This is particularly useful for rare event prediction, like machine failures.
* **Model Training Platforms:**
  + **Google Colab:** Utilize its free GPU resources for training complex models without incurring high computational costs.
  + **SageMaker Studio Lab:** Leverage Amazon’s free platform for a more controlled and scalable environment suitable for training and fine-tuning your models.

**2. Development of Predictive Maintenance Models:**

* **Vibration Analysis:** Develop models that can predict mechanical failures by analyzing vibration patterns from equipment.
* **Thermal and Acoustic Analysis:** Use temperature and sound data to predict overheating and mechanical wear, respectively.
* **Autonomous Decision-Making Algorithms:** Create algorithms that not only predict maintenance needs but also autonomously decide the best course of action, like scheduling maintenance or shutting down equipment temporarily to prevent damage.

**3. Implementation Considerations:**

* **Model Validation:** Regularly validate models against new and unseen data to ensure they continue to perform accurately over time.
* **Model Deployment:** Deploy models to edge devices using frameworks like TensorFlow Lite, which is optimized for low-power and low-latency operations on edge devices.
* **Feedback Loop:** Implement a feedback system where the model’s predictions and the actual outcomes are compared to further refine and improve the models.

**Documentation and Compliance:**

* **Documentation:** Keep detailed records of the model development process, including datasets used, model parameters, training and validation processes, and versioning.
* **Compliance and Standards:** Ensure that all AI models and data handling practices comply with relevant industry standards and regulations, particularly those relating to data security and privacy.

**3.Implementation and Testing**

**1. Model Deployment on Edge Devices:**

* **Selection of Deployment Tools:**
  + **TensorFlow Lite:** Use TensorFlow Lite to convert your AI models into a format that is optimized for low-power and low-latency execution on edge devices.
  + **Edge Impulse:** Leverage Edge Impulse for deploying machine learning models directly to edge devices, enabling real-time data processing and decision-making.
  + **MicroPython:** Utilize MicroPython for simpler control logic on smaller, less powerful microcontrollers that may be part of your IIoT environment.

**2. System Integration:**

* **Data Integration:** Ensure that data from various sensors (vibration, temperature, and sound) is seamlessly integrated and processed by the edge devices in real-time.
* **Autonomous Agent Activation:** Integrate the AI-driven autonomous agents to continuously monitor the system’s health and make decisions autonomously. This involves setting thresholds for alerts and actions based on the predictions from the AI models.
* **Real-Time Operation:** Set up the system to operate in real-time, ensuring that data processing, decision-making, and maintenance scheduling are performed without delays.

**Testing**

**1. Simulation Testing:**

* **System Simulations:** Use a simulated environment that mirrors the actual factory setup to test the entire system. This includes simulating sensor inputs, data transmission, edge processing, and response actions.
* **Focus Areas on Testing:**
  + **Real-Time Decision-Making:** Ensure the system can make and execute decisions in real-time based on the sensor data.
  + **Data Processing Efficiency:** Verify that data processing workflows are optimized and do not introduce unacceptable latencies.
  + **Autonomous Operations:** Test the autonomous agent’s ability to operate without human intervention under normal and faulty conditions.

**2. Security Testing:**

* **Threat Simulation:** Simulate various security threats, such as network intrusions, data tampering, and unauthorized access attempts, to assess the system’s resilience.
* **Vulnerability Assessments:** Use tools like penetration testing and vulnerability scanners to identify potential weaknesses in the system.
* **Mitigation Verification:** Ensure that security measures such as encryption, access controls, and anomaly detection are effectively protecting the system.

**2. Iterative Testing and Feedback:**

* **Continuous Integration/Continuous Deployment (CI/CD) Pipeline:** Implement a CI/CD pipeline to continuously integrate and deploy updates to the system, allowing for iterative improvements based on testing feedback.
* **Feedback Loop:** Establish a feedback loop with the system operators to continuously collect and integrate operational insights into system performance and user interface improvements.

**Documentation and Compliance:**

* **Testing Documentation:** Records of all tests performed, including the scenarios, methodologies, results, and any corrective actions taken.
* **Compliance Verification:** Testing procedures and the final system comply with relevant industry standards and safety regulations.

**4.Final Report Preparation**

* Project Goals and Objectives
  + The goals and objectives of this Capstone project are to create an AI integrated smart factory system “SmartGuard” that can monitor and predict maintenance on equipment with a low-cost edge device. The goals are to improve performance, reduce unexpected downtime and improve output on machines monitored by the SmartGuard system.
* System Design and Architecture
  + The system is based around low cost, easily obtained single board computers like the Nvidia Jetson, Orin and TX2. These boards, while low power, have many GPIO pins, I2C connectors and USB ports allowing many different sensors and peripherals to be connected. Using Ubuntu for the operating system, a system manager can easily set up PXE boot to allow the computer to boot from LAN, ensuring an up-to-date operating system and application package.
* AI Model Development and Deployment
  + The AI models used in this project are <https://www.kaggle.com/datasets/jishnukoliyadan/vibration-analysis-on-rotating-shaft> for a basis on rotational vibration analysis, as well as custom computer vision models for IR Temperature sensing, and custom models for thermocouple and audio analysis with the assistance of domain experts to assist in training. Models are deployed to the endpoints through TensorLite allowing for a small distribution package, helping to minimize the network load and improve the performance of the PXE boot system. This allows all endpoints to have the same model to ensure consistency across the platform.
* Implementation and Testing Results
  + Implementation is simple with a small device being placed on or near the monitored equipment and networking and or power provided to the device, depending on if PoE is used or not. Testing will determine optimal placement based on sensor configuration and other environmental factors. As the systems are running there are opportunities for fine tuning any false positives and false negatives to improve the dataset as it is exposed to the running environment.
* Security Measures and their Effectiveness
  + Standard enterprise security measures are in place, tamper proof/tamper evident casing, SSL encryption, data encryption at rest, the use of dedicated vLANs for endpoints and data traffic as well as firewalls between the endpoints and the backend system following a least access firewall configuration. The PXE boot from a secure server ensures that in the suspected compromise a system is wiped and loaded with a fresh OS each boot. All models and operating system distributions are checked against a known checksum to ensure that there is no modification made before deployment. IDS and IPS system check for unexpected traffic and shutdown any systems that are suspected of compromise. All inputs are bound checked to reduce the possibility of a buffer overflow, and the encryption of all traffic between endpoints prevents the use of Man in the Middle attacks.
* Reflections on the project, Challenges Face and Lessons Learned
  + Through this project we gained a much deeper understanding of what is involved in creating an enterprise ready Ai enabled system, looking at it from the aspects of deployment, security, testing and operating an environment to support such a system. Much was learned about the use of agents and IoT devices and their role in data acquisition and how to integrate them with an AI system. The understanding of how to ensure privacy, security, minimize attack surfaces, and reduce bandwidth consumption are crucial factors when designing a system like this.

**Presentation:**

Our presentation is on the SmartGuard AI based machine monitoring system for enabling Smart Factories. This system uses AI to understand the operation of a machine through environmental factors such as vibrational analysis, thermal changes, and sound analysis. These factors allow us to determine what is normal operating procedure and when there is a deviation from that. This allows for a factory that is more productive, has less unscheduled downtime and a higher output through better preventive maintenance planning.

Our system starts with low cost, powerful AI endpoints such as the NVidia Jetson, TX2 and Orin. This allows the sensor package to learn from your equipment and develop a baseline of what normal operation looks like. As the system learns, it will be able to determine when a system is operating in a way that does not meet that baseline and will generate alarms to notify of needed maintenance. All data is processed at the endpoint on hardware contained in tamper-proof housings, and only metadata is transmitted to the backend database. This minimizes the amount of data on your network as well as ensuring that no proprietary information could be sent across the wire. All data transfers are also SSL encrypted. After the metadata is aggregated in the backend database reporting is generated to show the status of all monitored systems viewable in a web-based dashboard, allowing for a single pane of glass overview of all equipment. In the event of alert SMS notifications are also sent out to alert teams responsible that a piece of equipment needs service.

Through this package we provide a way to bring artificial intelligence to your factory.