Use Case: AI & ML Agent for Predicting and Remediating Kubernetes Cluster Issues

Objective

This 45-day hackathon challenge invites participants to develop a model that can predict potential issues in Kubernetes clusters (Phase 1) and recommend or automatically implement solutions to address those issues (Phase 2). The aim is to optimize the efficiency of Guidewire solutions on Cloud Infrastructure. Participants can only leverage publicly available resources and Open Source solutions to achieve these goals.

Phase 1: AI/ML Model for Predicting Kubernetes Issues

Problem Statement

Kubernetes clusters can encounter failures such as pod crashes, resource bottlenecks, and network issues. The challenge in Phase 1 is to build an AI/ML model capable of predicting these issues before they occur by analysing historical and real-time cluster metrics.

Key Objectives for Phase 1:

Data Collection: Use publicly available datasets or simulate key metrics from Kubernetes clusters, such as CPU usage, memory usage, pod status, network IO, and other information that you may perceive to be relevant.

Model Design: Build a model capable of predicting issues mentioned below as a minimal viable scope (more can also be accommodated):

- Node or pod failures.
- Resource exhaustion (CPU, memory, disk).
- Network or connectivity issues.
- Service disruptions based on logs and events.

Prediction Accuracy: Focus on developing models that accurately forecast potential failures using techniques such as anomaly detection, time-series analysis, and other applicable techniques.

(Optional) Consume K8s: Package all dependencies in K8s to execute the solution.

Deliverables for Phase 1:

Build a Model: A trained machine learning model capable of predicting issues in Kubernetes clusters based on given or simulated data.

Codebase: Functional code including data collection, model training, and evaluation scripts uploaded to Github.

Documentation: Clear documentation explaining the approach, key metrics used, and model performance.

Presentation: A brief recorded presentation of the prediction model, including results and potential improvements together with a demo. Additionally, please upload the presentation file if applicable.

Test Data: Test data that was used for training and testing the model (If applicable).

Phase 2: Remediation for Predicted Issues

Problem Statement

Once issues are predicted, the next step is to automate or recommend actions for remediation. The challenge in Phase 2 is to create an agent or system capable of responding to these predicted issues by suggesting or implementing actions to mitigate potential failures in the Kubernetes cluster.

Key Objectives for Phase 2:

Remediation Actions: Based on predicted issues from Phase 1, develop a system that recommends or implements appropriate remediation steps. Examples include:

- Scaling pods when resource exhaustion is predicted.
- Restarting or relocating pods when failures are forecasted.
- Optimizing CPU or memory allocation when bottlenecks are detected.

Automation: Integrate the remediation system with the AI/ML agent to trigger automatic responses to predicted issues.

Evaluation of Effectiveness: Measure how effective the remediation actions are in mitigating or preventing cluster issues.

(Optional) Consume K8s: Package all dependencies in K8s to execute the solution.

Deliverables for Phase 2:

Remediation System: A functional system together with an agent that recommends the scripts to be run or automates remediation for predicted issues.

Codebase: Functional code implementing the remediation logic, connected to the Phase 1 prediction agent uploaded to Github.

Documentation: Detailed documentation describing how remediation actions are chosen or implemented.

Presentation: Final presentation of the complete solution, including both the prediction and remediation phases, with an emphasis on the integration of the two phases together with a recorded demo of the end-to-end process. Please upload the presentation files if applicable.

A Deployed Application: If possible, please deploy the application on a cloud platform of choice for us to try the agent live in action.

(Optional) Consume K8s: Package all dependencies in K8s to execute the solution.

Hackathon Duration & Timeline

Total Time: 45 Days

Phase 1 (Prediction)

- Recommended Duration: 1-20 days.
- Data Collection, Model Development, Training, and Evaluation.
- Submission of Phase 1 deliverables.

Phase 2 (Remediation)

- Recommended Duration: 21-45 days.
- Design and Development of Remediation Actions.
- Integration of Remediation System with Prediction Model.
- Submission of final deliverables.

Skills Required:

- Knowledge of Kubernetes and container orchestration.
- Machine learning (e.g., time series forecasting, anomaly detection).
- Experience with AI/ML libraries (e.g., TensorFlow, PyTorch, Scikit-learn).
- GenAI tools - LLMs, Langsmith/Langgraph or any other Open Source Solution.
- Python or relevant programming languages.
- Familiarity with Kubernetes APIs and monitoring tools (e.g., Prometheus).

Scoring Criteria (100 points max)

Ideation and Problem Understanding — 20 Points

- Identification of Key Failures, Data Utilization, Application of appropriate AI/ML techniques & Strategic Problem-Solving

Solution Excellence — 40 Points

- Solution Relevance & Accuracy
- Code Quality & Structure
- System Architecture
- Model Accuracy, Precision & Performance

Innovation & Creativity — 20 Points

- Novelty of Approach
- Overcoming Challenges

Demonstration & Presentation — 20 Points

- User Experience & Accessibility
- Documentation Quality & Clarity
- The Presentation and Demonstration of the solution
- Creativity Factor (X-factor)

Resources:

- Kubernetes Documentation: Kubernetes Official Docs
- Prometheus Documentation: Prometheus Monitoring
- Grafana Documentation: Grafana
- AI/ML Tutorials:
  - Scikit-learn Documentation
  - TensorFlow Documentation
  - PyTorch Documentation
- AI Agents:
  - Agents
  - Introducing ambient agents