UNIVERSITY OF MORATUWA

Faculty of Engineering

Registered Module No: EN3993

INDUSTRIAL TRAINING REPORT

Information Systems Associates (Pvt) Ltd.

From: 04/12/2023 To: 20/05/2024

Date of Submission:

31/05/2024

Y.U.K.K. Chandrasiri

200087A

Department of Electronic and Telecommunication Engineering

Preface

This report details my internship as a trainee - software engineering, from December 04th, 2023, to May 20th, 2024, at Information Systems Associates (Private) Limited, a Company duly incorporated under the laws of Sri Lanka having its company no. PV96998 at Level 02, Prince Alfred Tower, No. 10, Alfred House Gardens, Colombo 3, Sri Lanka. Hereinafter referred to as the “Company" or “ISA” ON THE ONE HAND. This internship was an essential part of the EN3993 module, "Industrial Training," in the BSc. (Hons) in Electronic and Telecommunication Engineering program at the Faculty of Engineering, University of Moratuwa. It served as a crucial link between theoretical knowledge and practical application, significantly contributing to my professional growth in the IT industry.

The report is structured into three main sections. The first section provides an overview of ISA, encompassing its background, organizational structure, primary industries, products, and services. Additionally, it includes a SWOT analysis of the company and recommendations for enhancements derived from my internship experience and observations.

The second section delves into my internship involvement, elucidating the production processes, administrative procedures, and office protocols at ISA. It also delineates the projects I engaged in and underscores my contributions to their accomplishments. Furthermore, it addresses the challenges encountered and the strategies employed to surmount them.

The third section encapsulates my overall training encounter. It furnishes feedback on the quality of training imparted by ISA, evaluates the efficacy of the internship program administered by the faculty, and provides insights into the training personnel at ISA. Conclusively, it offers recommendations regarding ISA's suitability for prospective interns, based on firsthand experience.

Through this comprehensive report, I aim to encapsulate and scrutinize my internship voyage at ISA, showcasing the invaluable insights garnered, the hurdles surmounted, and recommendations for continual enhancement. This experience has been pivotal in sculpting my professional trajectory, and I trust this report will furnish pertinent guidance for aspirants seeking akin opportunities.

Acknowledgment

I extend my sincere gratitude to the Industrial Training Division of the University of Moratuwa, the Department of Electronic and Telecommunication Engineering at the University of Moratuwa, the National Apprentice and Industrial Training Authority, and ISA for affording me this invaluable opportunity.

I am deeply thankful to Eng. Roy Sankaranarayana, Director of the Industrial Training Division, and the entire team for orchestrating the industrial training program. My gratitude also extends to Dr. Ranga Rodrigo, the former Head of the Department of Electronic and Telecommunication Engineering, Dr. Subodha Charles, the Training Coordinator, and all the academic staff of the department for their steadfast guidance and support during my 24 weeks of industrial training.

A special appreciation is reserved for my supervisor and the Head of the Department, Mr. Dilan Anuruddha, alongside Mr. Rimaz Fawzer, the team leader, and Mr. Ravindra Singh and Mr. Manoj Nilanga, my mentors, for bestowing upon me this opportunity and furnishing invaluable guidance throughout my internship at ISA.

This opportunity has been pivotal in shaping my professional journey, and I express heartfelt gratitude to all individuals and organizations whose contributions made this experience possible.

Table of Contents

List of Figures

List of Tables

1. Introduction to the Training Establishment

1.1. Basic Details

Figure 1; Company Logo

Table 1: Company Details

Name	Information Systems Associates (Private) Limited
Address	Level 02, Prince Alfred Tower, No. 10, Alfred House Gardens, Colombo 3, Sri Lanka.
Contact	+94 112058886 info@isa.ae
Official Website	isa.ae
Founded Year	2005

1.2. Story

ISA (Information Systems Associates) is a premier in the field of Information Technology providing best-of-breed technology solutions for the global travel and aviation industry since 2005. It offers a wide range of tailor-made aviation technology solutions which cater to every aspect of the organization. It has proven its expertise and proficiency over the years and have attracted scores of prestigious clients around the world. It is headquartered in Sharjah – UAE, while the Research and Development centre is located in Colombo, Sri Lanka.
Since making its debut with ACCELaero, a modern online reservation system, it has strongly evolved adopting cutting-edge technology to meet the ever-growing demands of the aviation sector today.

Today ISA has a wide range of products which are catered to the Airline, Airport, Travel and Tourism sectors, it takes great pride in saying that it has made remarkable progress since its inception, reaching many milestones along the way, as well as setting ambitious plans for its future.

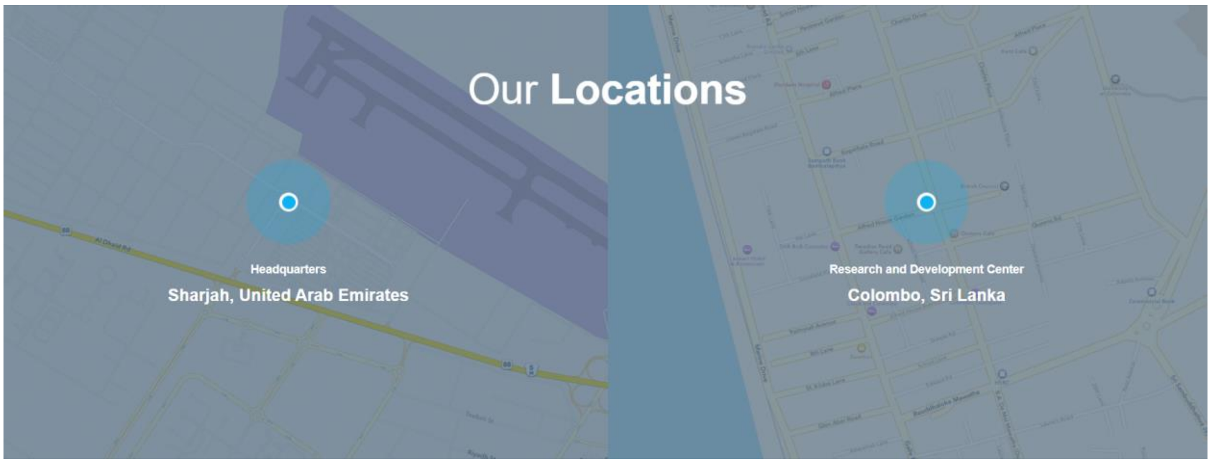
Our Locations
Headquarters
Sharjah, United Arab Emirates
Research and Development Center
Colombo, Sri Lanka

Figure 2: Global Presence

1.3. Vision & Mission

Engineered for Success

The leading IT provider in the aviation industry. specialize in delivering cutting-edge aviation software.

1.4. Company values

Table 2: Company Values

Innovation and Excellence	“At ISA, we are committed to delivering cutting-edge aviation software solutions that set new benchmarks in the industry. We strive for excellence in everything we do, ensuring that our products and services consistently exceed client expectations.”
Transparent Communications	“We believe in maintaining open and honest communication with our clients. By providing regular updates and accurate information, we ensure that our clients are always informed and involved in their projects, fostering trust and satisfaction.”

Commitment to Continuous Improvement	"Our dedication to continuous learning and improvement drives us to constantly enhance our skills and knowledge. We take personal responsibility for our actions and commitments, ensuring that we deliver predictable and high-quality outcomes for our clients."
Client-Centric Approach	"At ISA, we prioritize our clients' needs and goals. By combining our expertise with creative thinking, we develop innovative solutions that provide real value. We are committed to delivering customized, high-quality business solutions that drive our clients' success."
Social Responsibility and Inclusivity	"We are dedicated to making a positive impact on society and promoting equal opportunities. Our diverse team brings together a wealth of perspectives and ideas, which we leverage to benefit our clients and the wider community. We believe in fostering an inclusive environment where everyone is valued and can contribute to our shared success."
Professional Integrity	"Maintaining high standards of personal and professional integrity is paramount at ISA. We adhere to rigorous ethical practices and corporate governance, ensuring confidentiality and fostering a culture of trust and respect within our team and with our clients."

1.5. Achievements

Being around 20 years in Airline Industry, ISA and its products attained remarkable success, earning multiple awards and accolades for its outstanding achievements.

- ISA was awarded "World's Best PSS Provider 2023"
- JC (Cambodia) International Airlines selects ACCELAERO
- accelAERO CEO wins ASIA PACIFIC'S MOST PRESTIGIOUS AWARDS FOR ENTREPRENEURS in 2017
- ISA has crowned a momentous by being named as World's Leading PSS Provider – 2016
- ISA signs contract with Wataniya Airways in Kuwait to power the airline's Passenger Service System with ACCELaero02 October – 2016
- ISA bags two accolades at 2016 Business Destinations Travel Awards
- ISA Named "World's Leading Passenger Service System Provider 2015"
- aeroMART SELL – First CRS in the world to get certified on PNRGOV for UAE and KSA – 2015

1.6. IT Services

- Network Management
- Cybersecurity Solutions
- Cloud Infrastructure Management
- IT Advisory and Consulting

1.7. Key Industries

- Aviation Industry
- IT Industry
- E-commerce

1.8. Products

In this section, I highlight various projects and products developed by ISA, showcasing the company's diverse expertise and innovative solutions.

1.8.1. aeroOPS

Cost effective aircraft utilization without compromising safety and efficiency.

About:

In the fast-paced world of aviation, airlines must have a keen eye on their end-to-end processes. That's where a reliable airline operation control system comes into play. Introducing aeroOPS, the remarkable solution that empowers airlines to effortlessly track, manage, and swiftly respond to their daily operational needs. aeroOPS provides the OCC team with comprehensive visibility, efficient planning capabilities, and real-time updates for streamlined flight operations.

Using aeroOPS an airline can easily plan all short-term and long-term tail assignments. Our solution effectively ensures aircraft scheduling helps to maximize profitability while minimizing downtime and maintenance requirements.

The ML-powered aeroOPS Co-Pilot is a powerful tool that helps controllers focus on the information that is most important. It declutters and consolidates information into a single, focused view, so you can see what you need on demand.

Elements:

- Flight Planning
- Flight Tracking
- Reporting
- ML-powered Co-Pilot

Key-features:

- Optimizing fleet utilization for maximum efficiency
- Revolutionize your aircraft allocation with optimized tail assignments
- Experience seamless disruption management in real-time, optimizing efficiency
- Unlock the power of complete operational visibility
- Real-time flight updates for proactive decision-making

1.8.2. aeroPORT

Designed to keep up with the latest technology and security standards.

About:

aeroPORT stands as the ideal Departure Control System (DCS), embraced by airports, airlines, and ground handlers as the leader in passenger handling professionally. This modern solution seamlessly exceeds the expectations offering flawless access from desktop computers to mobile devices.

aeroPORT amalgamates vital and fundamental elements of a DCS installed in airports across the globe. Our platform embodies the essence of internationally endorsed standards and best practices. Our software executes operations with unwavering reliability while preserving the key aspects of security and user-friendliness.
aeroPORT integrates with all airport management systems orchestrating departures that are both efficient and secure on a global scale, catering to the discerning needs of airlines and airports no matter the location.

Elements:

- Secure and Compliant
- Real-time Data Availability
- Key Integrations Supported
- Web-based Platform

Key-features:

- Operational simplicity
- Provides government messaging
- Boarding card & bag tagging
- Printer agnostic

1.8.3. aeroMART

Traveler-centric retailing, Transformed.

About:

With aeroMART, airlines can effortlessly incorporate new features and services, all while boosting their agility and responsiveness without the need for an extensive system overhaul. As a microservice-based Passenger Services System (PSS) airlines can reimagine the way passengers are managed. New features can be easily added without disrupting the entire system. Each module can function independently thus guaranteeing minimal disruptions in case of updates or maintenance.
aeroMART optimizes ancillary revenue opportunities by offering upselling and cross-selling capabilities on par with modern retailing platforms from any industry. Known for stable APIs and developer-friendly documentation we make it easy for airlines to extend and customize the distribution.
Flexibility and scalability are core to our solution and mission to enable airlines to reduce operational costs while improving resource utilization. aeroMART is more secure than traditional monolithic platforms.

Elements:

- Inventory management & Pricing
- Direct and Indirect Distribution
- Modular structure
- Cloud Based
- OTA Connections

Key-features:

- aeroORDER
- aeroXBE
- aeroAGENT
- aeroINVENTORY
- aeroANCI
- aeroPRICE
- aeroTAX

1.8.4. aeroLINE

Technology committed to crew well-being and productivity.

About:

Efficient crew management is the backbone of any successful airline operation. At ISA, we recognize the intricate challenges that come with managing airline crew. That's why we have developed aeroLINE, a comprehensive crew management system that goes beyond being a mere operational tool. It serves as a robust platform, empowering operational teams, crew members, and stakeholders to strike the perfect balance between crew productivity, crew utilization, crew satisfaction, operational efficiency, and cost optimization.
aeroLINE Crew Portal and Mobile Applications: Empowering Crew with Seamless Management
aeroLINE-Crew offers a versatile and device-agnostic crew portal and mobile applications that put crew members in control of their schedules and streamline essential functions like check-in, bidding for duties, expiry tracking, etc. The intuitive platform provides real-time access to important notifications, including roster changes, qualification expiries, memos, crew swaps, and operational alerts while on the move. Additionally, it ensures uncompromised data layer security to bestow confidentiality of operational data.

Elements:

- Pairing Management and Optimization
- Roster Management and Optimization
- Holistic Training Management
- Crew Communication
- Reporting & Analytics

Key-features:

- Unlock the power of perfect pairings for your roster
- FDP excellence made easy
- Efficient equitable crew duty distribution
- Eliminate delays & cancellations
- Streamlined training & qualification management solution
- Streamlined leave requests and enhanced schedule control

1.9. Clients and business partners

The main business partner of ISA is Air Arabia Group.

AirArabia
العربية للطيران

Figure 3: Business partner

1.10. Organizational structure of the company

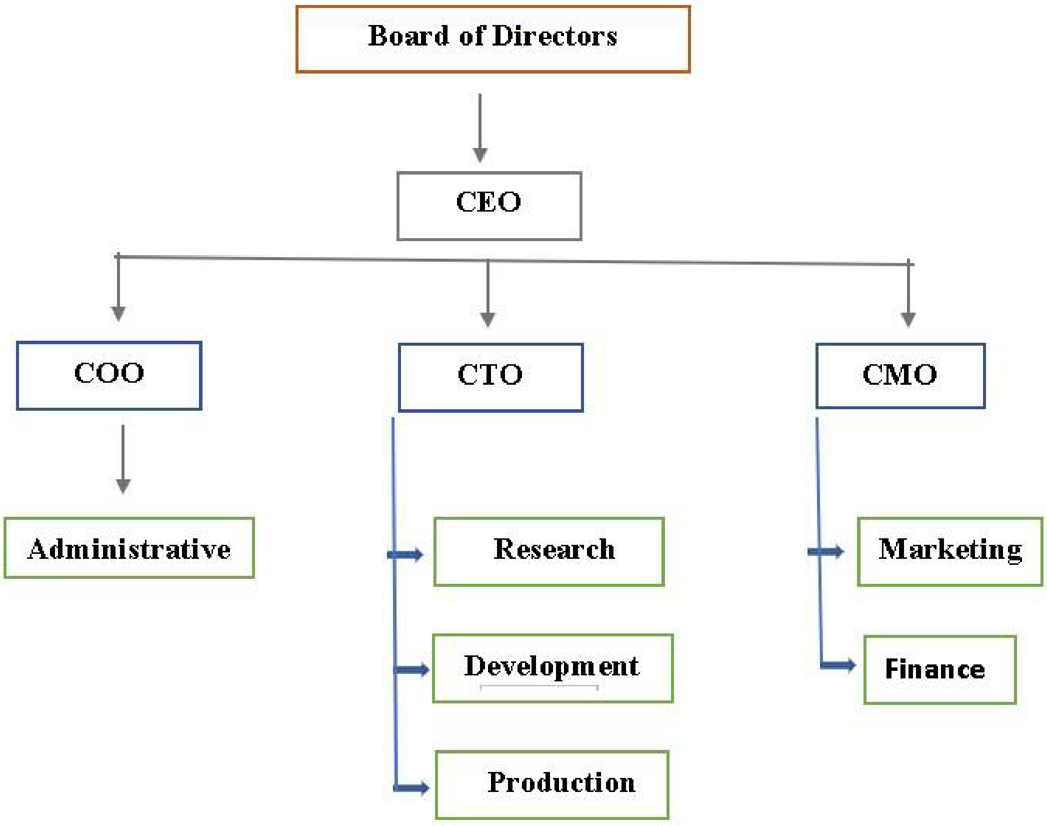

Figure 4: Organizational structure of the company

1.11. SWOT analysis of the Training Establishment’s present performance

The SWOT Analysis serves as a strategic tool to assess an organization by scrutinizing its Strengths, Weaknesses, Opportunities, and Threats. This method involves an internal examination to identify strengths and weaknesses, coupled with an analysis of the external landscape to uncover potential opportunities and threats. Through conducting a SWOT analysis, organizations can gain valuable insights into their present standing, enabling them to make informed decisions to leverage strengths, mitigate weaknesses, seize opportunities, and address threats effectively.

1.11.1. Strengths

- Recognized for award-winning products.
- Supported by a team of aviation domain experts.
- Competitive pricing strategy.
- Offers customizable solutions.
- Flexible on-demand delivery system.
- Adheres to ISO Certified Quality Standards.
- Operates from a PCI DSS Compliant Data Center.
- Emphasizes UX-driven solution designs.
- Implements agile practices and methodologies.

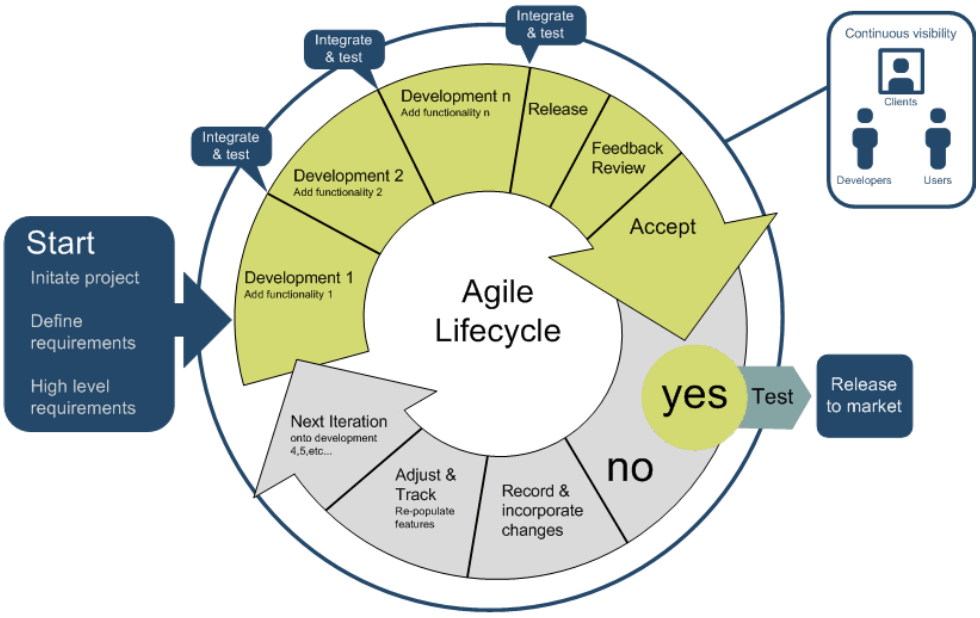

Figure 5: Development process of the company

1.11.2. Weaknesses

- Company culture leans heavily towards work, lacking relaxation spaces within premises.
- Social media presence is minimal, hindering access to the latest information and limiting publicity.
- Insufficient activities to foster teamwork and develop soft skills among employees.

1.11.3. Opportunities

- Strong partnership with the Air Arabia group provides a secure and vast array of job opportunities.
- Commitment to staying abreast of emerging technologies, such as LLM and advanced AI, through initiatives like Hackathons and webinars positions ISA to pioneer future products and services.
- Ability to offer services both locally and internationally enhances credibility and opens doors for expansion and growth.

1.11.4. Threats

- Intense competition within the IT industry poses a challenge, with numerous companies and startups vying for market leadership.
- The competitive landscape presents a potential threat to ISA's market position and growth trajectory.

1.12. Suggestions

Drawing from my internship experience, I present the following suggestions for ISA:

1. Diversify Team Roles

In order to achieve a balanced workload and ensure successful completion of various stages within the software development lifecycle, ISA should contemplate diversifying team roles. This can entail assigning roles such as QA engineers and developers across teams, rather than consolidating them into separate entities. Such diversification will optimize resource allocation and enhance efficiency in project execution.

2. Enhance Social Media Presence

To remain competitive and attract top-tier talent, ISA should actively bolster its presence on social media platforms. Consistent engagement through regular updates and interactions will not only increase the company's visibility but also fortify its position in the market.

3. Establish a Dedicated HR Department

To address inefficiencies and streamline HR-related processes, ISA should consider establishing a dedicated HR department. This dedicated unit can facilitate seamless onboarding procedures, promptly address HR-related matters, and ultimately elevate overall employee satisfaction.

4. Prioritize Employee Development

Investing in the professional development of employees is paramount for ISA's growth and success. By organizing workshops, providing access to online courses, and fostering a culture of continuous learning, the company can equip its workforce with the latest industry knowledge and technologies. Additionally, creating an environment that promotes relaxation for developers will contribute to their overall well-being and productivity.

2. Internship experience

2.1. Overview

After successfully navigating both a technical interview and an HR interview, I was extended an offer for the position of Trainee Software Engineer at ISA. My internship tenure spanned from December 4th, 2023, to May 20th, 2024. Upon commencing my internship, I was tasked with collaborating alongside the DevOps team within the company.

In the inaugural week, the HR department orchestrated an orientation session, furnishing invaluable insights into the company's background, product portfolio, corporate culture, and requisite policies and guidelines. Subsequently, during the monthly status update meeting, I, along with other new recruits, had the opportunity to introduce ourselves to both the team and the broader company.

Throughout the initial weeks, I was provided with an array of resources and materials geared towards augmenting my proficiency in Oracle Cloud Infrastructure, Docker, Kubernetes, and HELM. This initiative aimed to furnish me with a comprehensive understanding and hands-on experience pertinent to the forthcoming project domain.

2.1.1. Production processes, administrative and office practices

The internship at ISA was primarily conducted in-person, necessitating all trainees to work from the office premises. While permanent staff were granted the option of remote work post-COVID-19 pandemic, trainees could seek approval for remote work with valid justifications. Daily stand-up scrum meetings with supervisors were facilitated through MS Teams, offering convenient access to guidance and assistance. Each day, we diligently logged JIRA entries, furnishing comprehensive reports on our progress regarding assigned tasks. The company embraced flexible working hours, fostering collaboration with supervisors to optimize work arrangements.

In the initial phase, I benefited from invaluable tutorials on OCI, Docker, Kubernetes, and engaging Knowledge Transfer (KT) sessions highlighting Agile best practices. However, in compliance with company policy, specific development details such as code snippets or diagrams cannot be disclosed.

For project development, the Agile Scrum methodology formed the cornerstone, meticulously tailored to accommodate the company's resources and circumstances. Daily SCRUM meetings held at 11:45 am facilitated team members in sharing updates, strategizing tasks for the day, and addressing potential challenges.

Furthermore, status update meetings ensued after every project, featuring active involvement from all members of the DevOps team. Each development team presented the status of their respective projects, providing newcomers with a platform to introduce themselves. Additionally, HR meetings encompassed company announcements, farewell gatherings for departing colleagues, and engaging recreational activities fostering camaraderie and enjoyment.

To support new hires, Knowledge Transfer (KT) sessions were conducted by seasoned team members. These sessions proved immensely beneficial, particularly for newcomers like myself, offering insights into diverse technologies and fostering opportunities for forging new connections.

2.1.2. Problems and difficulties encountered, and solutions found

Initially, adjusting to virtual supervision posed a considerable challenge for me. Whenever I encountered obstacles, I relied on reaching out to our supervisor for guidance. Thankfully, there was always a supportive network of individuals available to lend assistance whenever issues arose. As I delved deeper into unfamiliar territory, I was provided with a plethora of resources and connected with seasoned professionals who graciously shared their expertise.

Beyond refining my technical acumen, this experience also served as a catalyst for significant improvement in my soft skills. In the nascent stages, I grappled with a lack of confidence when confronted with novel technologies. However, as time progressed, my aptitude for learning sharpened. Through hands-on immersion, I cultivated a greater sense of assurance in problem-solving and navigating uncharted domains.

Moreover, collaborating within a team dynamic enriched my capabilities as a team player. I found myself actively assisting others and engaging with new team members, thereby augmenting my social and communication proficiencies. Thanks to this immersive experience, I now exude a heightened sense of confidence and proficiency in seamlessly operating within a collaborative team environment.

2.2. Projects involved.

2.2.1. Alert Pilot – Prometheus Alert Enrichment System

2.2.1.1. Introduction

Figure 6: Keep Logo

What is Kubernetes Cluster

A Kubernetes Cluster is a set of nodes that run containerized applications managed by the Kubernetes platform. It consists of a master node and multiple worker nodes. The master node controls and manages the entire cluster by handling the orchestration of the containers, scheduling deployments, and scaling applications. Worker nodes are responsible for running the actual applications in containers. Kubernetes clusters enable developers to efficiently deploy, manage, and scale applications across a distributed infrastructure, providing high availability, fault tolerance, and streamlined management of microservices.

What is Alert

An alert is an event that is triggered when something undesirable occurs or is about to occur. It is usually triggered by monitoring tools such as Prometheus, Grafana, or CloudWatch, and in some cases, proprietary tools.

Alerts usually categorized into three different groups:

- Infrastructure-related alerts - e.g., a virtual machine consumes more than 99% CPU.
- Application-related alerts - e.g., an endpoint starts returning 5XX status codes.
- Business-related alerts - e.g., a drop in the number of sign-ins or purchases.

Prometheus

Prometheus is an open-source monitoring and alerting toolkit designed for reliability and scalability. It collects metrics from various endpoints, stores them in a time-series database, and provides a powerful query language called PromQL for real-time data analysis. Prometheus is widely used for monitoring system performance, applications, and infrastructure. It operates by periodically scraping metrics from instrumented services, storing them efficiently, and enabling users to visualize and alert on the data through integrations with tools like Grafana. Its flexibility and rich feature set make it a popular choice for monitoring in modern cloud-native environments.

This is how it is implemented in our system.

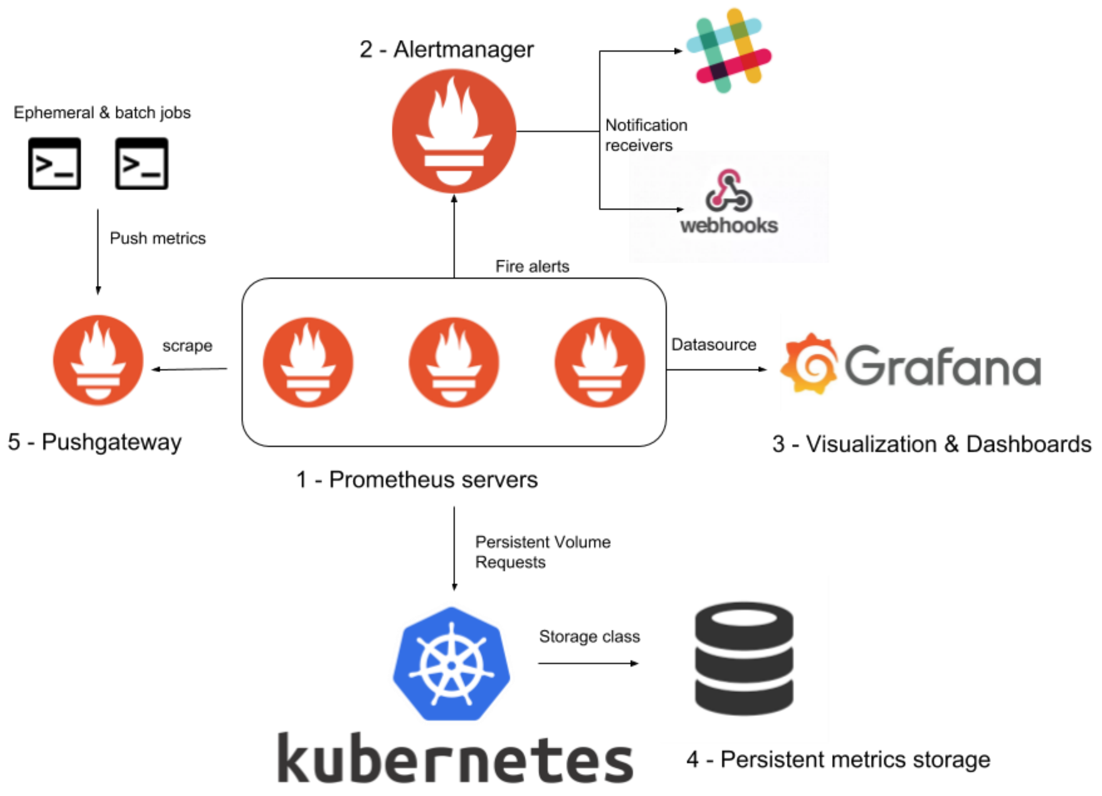

Figure 7 : How Prometheus Alert Manager Works

Prometheus Alert Manager

Prometheus Alertmanager is a component of the Prometheus ecosystem responsible for handling alerts generated by Prometheus. It receives alert notifications, processes them according to user-defined rules, and sends them to designated receivers such as email, Slack, or PagerDuty. Alertmanager supports grouping, deduplication, and silencing of alerts to reduce noise and ensure that notifications are meaningful and actionable. It helps organizations maintain robust alerting workflows, ensuring timely responses to critical issues and facilitating efficient incident management in dynamic and complex environments.

Alert Enrichment

The information in alerts sent using Prometheus Alertmanager under defined alert rules is often insufficient for developers and DevOps team to make quick decisions. As a result, the relevant personnel must execute commands manually to determine the exact cause of a particular alert. To automate this time-consuming task, alert enrichment can be employed. This involves running an automated process based on the alert type to gather additional details, which are then used to enhance the alert information before sending it back to the Alertmanager. This approach streamlines the troubleshooting process and allows for more informed and efficient responses to alerts.

Keep by KeepHQ

Keep is an open-source alert management and automation tool that provides everything we need to create and manage alerts effectively.

Keep helps with every step of the alert lifecycle:

- **Creation -** Keep offers a framework for creating, debugging, and testing alerts through code that scales with your teams.
- **Maintenance** - Keep integrates with your tools, allowing you to manage all of your alerts within a single interface.
- **Noise reduction** - By integrating with monitoring tools, Keep can deduplicate and correlate alerts to reduce noise in your organization.
- **Automation -** Keep Workflows enable automated alert enrichment and response.

This is how Keep workflow works:

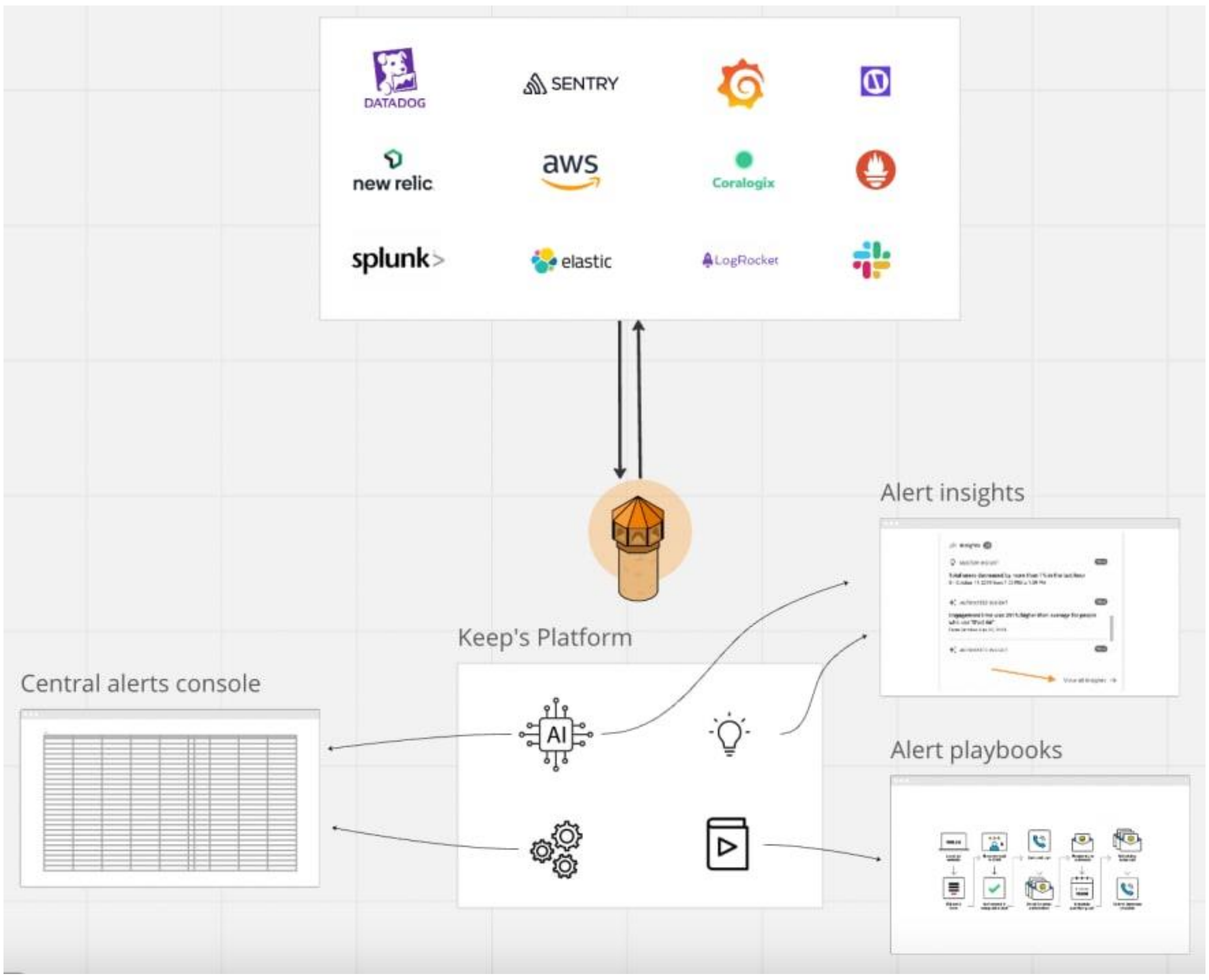

Figure 8 Keep Workflows

2.2.1.2. My Contribution

2.2.1.2.1. Overview

The Alert Pilot project aims to enhance Prometheus alerts originating from the company's Kubernetes cluster using Keep/KeepHQ. This involves automating a process based on the alert type to gather additional details, enriching the alert information before forwarding it to the Alertmanager. This method optimizes the troubleshooting process, allowing for more informed and efficient responses to alerts.

Technology Stack:

- **Tools**: Prometheus Alertmanager, Keep Alert Management System
- **Frameworks**: Kubernetes Python Client, Python Prometheus API
- **Scripts**: Python Scripts, Keep Workflows (YAML)
- **Environment**: Kubernetes Cluster in OKE (Oracle Kubernetes Engine)

2.2.1.2.2. Implement Scripts using Kubernetes Python Client

In this phase, I developed the necessary Python scripts to be executed based on the received alert types. These scripts utilize the Kubernetes Python Client to enrich alert information by gathering additional details from the Keep workflow and sending the enhanced alerts back to the Alertmanager. The scripts are designed to streamline the troubleshooting process by providing more comprehensive information for each alert type.

The scripts implemented are as follows:

- **Handle_elastic_alerts.py**: This script handles alerts of types "HostHighCpuLoad", "HostOutOfDiskSpace". and "HostHighCpuLoadP1". It checks the health of Elastic Datastreams and enriches the alert with this information before sending it back to the Alertmanager.
- **Get_crashed_container_logs.py**: For alerts of type "KubePodCrashLooping", this script identifies the crashed pod and container, retrieves the logs of the crashed container, enriches the alert with these logs, and sends it back to the Alertmanager.
- **Get_pod_events.py**: This script deals with alerts of type "KubePodNotReady". It retrieves the pod in the not-ready state, gathers the pod's events, enriches the alert with these events, and sends it back to the Alertmanager.
- **Handle_istio_5xx_alerts.py**: For alerts of type "IstioHighErrorRateCorelation", this script identifies the service with a high error rate, checks the dependent services for similar issues, enriches the alert with this information, and sends it back to the Alertmanager.

2.2.1.2.3. Implement Keep workflows to automate enrichment

In this phase, I developed Keep workflows to automatically run upon receiving specific alerts from the Prometheus Alertmanager. A workflow in Keep is a YAML-based configuration file designed to manage, automate, and enrich alerts. Once uploaded to Keep, the workflow can be triggered manually, by an alert, or at set intervals. Keep collects all alerts based on the alert rules, passes the alert details to the Python scripts, and executes them according to the alert type.

Keep workflows consist of four main parts:

- **Triggers** - A trigger is an event that starts the workflow. It could be a manual trigger, an alert, or an interval depending on your use case. Keep support three types of triggers.
- **Steps** - Steps are optional and define a sequence of actions that fetch or compute data. They are used to add data to the workflow, which can be used in other steps or actions.
- **Actions** - An action defines what to do when a workflow is triggered. Actions usually rely on providers for executing specific tasks, like sending a Slack message.
- **Conditions** - A condition sets the rules under which an action should be performed. For example, you can set a condition to only trigger an action if certain criteria are met.

The keep workflows I implemented are as follows.

- ElasticHostAlerting.yaml
    - Trigger: r"(HostHighCpuLoad|HostOutOfDiskSpace|HostHighCpuLoadP1)"
    - Steps: run “handle_elastic_alerts.py {{ alert }}”
- IstioHighErrorRateCorelation.yaml
    - Trigger: r"(HighHTTP5xxReportedByClientP1|HighHTTP5xxReportedByClient)"
    - Steps: run “handle_istio_5xx_alerts.py {{ alert }}”
- KubePodCrashLoopings.yaml
    - Trigger: (KubePodCrashLooping)
    - Steps: run “getlogsofcrashedpods.py {{ alert }}”
- KubePodNotReady.yaml
    - Trigger: (KubePodNotReady)
    - Steps: run “getevents.py {{ alert }}”

2.2.1.2.4. Research on Istio Service Dependencies

The objective of this phase was to utilize Istio's capabilities to map out service dependencies within our company's Kubernetes cluster.,

What is Istio

Istio is an open-source service mesh that provides a uniform way to connect, secure, control, and observe microservices. It helps manage the communication between microservices deployed within Kubernetes clusters by providing tools for:

- **Traffic Management**: Istio allows fine-grained control of traffic behavior with rich routing rules, retries, failovers, and fault injection.
- **Security**: It offers end-to-end authentication, authorization, and encryption of service communication.
- **Observability**: Istio provides detailed insights into service behavior through monitoring and tracing, enabling the identification of issues and bottlenecks.
- **Policy Enforcement**: It supports fine-grained control over service behavior with access controls and rate limiting.

Objective

The main goal was to create a mechanism that identifies all destination services interacting with a given source service within our Kubernetes cluster. This information is crucial for understanding service dependencies.

Methodology

1. Data Collection:
   - Leveraged Istio’s telemetry features to gather data on service interactions.
   - Used Istio’s Prometheus integration to collect metrics on service traffic.
2. Script Development:
   - Developed Python scripts using the Kubernetes Python Client and Istio APIs.
   - These scripts query Istio’s telemetry data to extract information about service interactions.

3. Alert Enrichment:
   - Import the Scripts to necessary scripts to get dependent services
   - Filtered and processed data to ensure accuracy and relevance.

2.2.1.2.5. Research on Kubernetes Troubleshooting

Introduction

This phase aimed to investigate and understand the manual troubleshooting processes for common Kubernetes issues before automating these processes. By gaining hands-on experience with manual troubleshooting, we aimed to develop a deeper understanding of the issues and how best to address them with automated solutions.

Kubernetes Troubleshooting: Key Scenarios

1. KubePodNotReady:
   - Issue: A pod is in a "Not Ready" state.
   - Troubleshooting Steps:
     - Check Pod Events: Retrieve the events associated with the pod to identify why it is not ready.
     - Look for events indicating issues such as image pull errors, insufficient resources, or failed probes.
     - Check Pod Status: Examine the status of each container within the pod.
     - Identify if containers are in Waiting, Running, or Terminated states and investigate accordingly.
2. KubePodCrashLoopBackOff:
   - Issue: A pod is in a CrashLoopBackOff state, indicating that a container within the pod is repeatedly failing.
   - Troubleshooting Steps:
     - Identify the Crashed Container: Determine which container is crashing.
     - Look for the container that has a Last State of Terminated and a Reason indicating a crash.
     - Read Container Logs: Retrieve and inspect the logs of the crashed container to understand the cause of the failure.

        - Analyze the logs for errors or stack traces that point to the root cause of the crash.

3. KubePodImagePullBackOff:
    - Issue: A pod is unable to pull its container image.
    - Troubleshooting Steps:
        - Check Pod Events: Review events to understand why the image pull is failing.
        - Look for events related to Failed to pull image or ImagePullBackOff.
        - Verify Image Name and Tag: Ensure that the image name and tag are correct and accessible from the container registry.
        - Check Registry Access: Confirm that Kubernetes nodes have access to the container registry and that credentials (if required) are correctly configured.
4. KubePodOOMKilled:
    - Issue: A container in a pod has been killed due to an out-of-memory (OOM) condition.
    - Troubleshooting Steps:
        - Check Pod Status: Identify which container was OOMKilled.
        - Look for State and Last State showing OOMKilled.
        - Examine Resource Limits: Verify the resource requests and limits set for the container.
        - Ensure that the container has appropriate memory limits and consider increasing them if necessary.
        - Review Container Logs: Check the logs leading up to the OOM kill for clues.

Conclusion

The manual troubleshooting processes for Kubernetes issues such as KubePodNotReady, KubePodCrashLoopBackOff, KubePodImagePullBackOff, and KubePodOOMKilled involve detailed investigation of pod events, container statuses, logs, and resource configurations. Understanding these processes is essential before automating the troubleshooting workflow. By gaining hands-on experience, we can create more effective and accurate automated scripts and workflows, ensuring that our Kubernetes clusters run smoothly and efficiently.

2.2.1.2.6. Testing & Deployment

After developing the system, I proceeded to deploy it to the Kubernetes staging cluster for thorough testing. The testing phase involved simulating various alert scenarios to ensure the system responded correctly and efficiently. This phase was crucial for identifying and resolving any issues before the system went live.

Deployment to Staging Cluster

1. Preparation:
   - Prepared the deployment manifests and scripts for the staging environment.
   - Ensured that the staging cluster mirrored the production environment to accurately test the system's performance.
2. Deployment:
   - Deployed the system components, including the Python scripts, Keep workflows, and necessary configurations, to the staging cluster.
   - Configured Prometheus and Alertmanager in the staging environment to generate and manage alerts for testing purposes.
3. Testing:
   - Conducted extensive testing by generating different types of alerts such as KubePodNotReady, and IstioHighErrorRateCorrelation.
   - Verified that the system correctly identified the alert type, executed the appropriate script, enriched the alert with additional information, and sent it back to the Alertmanager.
   - Monitored the logs and events to ensure that the system performed as expected without errors or performance issues.
   - Iteratively refined the system based on test results, addressing any bugs or inefficiencies encountered.

Deployment to Production Cluster

1. Code Review and Pull Request:
   - After successful testing, I prepared the final version of the system for production deployment.
   - Opened a pull request in Bitbucket, detailing the changes, testing outcomes, and deployment instructions.

    - The pull request was reviewed by my supervisor and other team members to ensure code quality, security, and compliance with best practices.
2. Approval and Merge:
    - Addressed feedback from the code review, making necessary adjustments and improvements.
    - Upon approval, the pull request was merged into the main codebase.
3. Production Deployment:

- My supervisor deployed the system to the production Kubernetes cluster following the established deployment procedures.
- Ensured minimal disruption to ongoing operations by coordinating the deployment during a scheduled maintenance window.
- Monitored the system closely post-deployment to ensure it functioned correctly in the production environment and to quickly address any unforeseen issues.

Conclusion

Deploying and testing the system in the staging cluster was a critical step in validating its functionality and reliability. The thorough testing process ensured that the system was robust and ready for production use. The deployment to the production cluster, overseen by my supervisor, marked the successful completion of the project. This deployment strategy ensured that the system was rigorously tested and securely transitioned to production, thereby minimizing risks and ensuring a smooth operational experience.

.

2.2.1 PostgreSQL Database Encryption

Project: How to encrypt data in PostgreSQL database in server using pg-crypto. Here a sensitive data column in a Database Table is encrypted using symmetric and as well as asymmetric encryption.

Technology Stack:

- Database: PostgreSQL
- Tools: gnupg , pg-crypto, PGAdmin4 , Flask API
- Concepts used: Symmetric Encryption, Asymmetric Encryption , RSA
- Languages: python3
- Containerization tools and APIs: Docker, docker-compose

Link: kavindukalinga/PostgreSQL-Database-Encryption-using-Pgcrypto: This repository contains how to encrypt data in PostgreSQL database in server using `pgcrypto`. Here a sensitive data column in a Database Table is encrypted using symmetric and as well as asymmetric encryption. (github.com)

2.2.2 Confluence Bot

Project: An NLP project aimed at crafting a chatbot capable of answering questions sourced from provided documents. It leverages open-source large language models (LLM) and retrieval augmented generation (RAG) techniques for this purpose.

Technology Stack:

- Tools: ollama, langchain
- Concepts used: Natural Language Processing (NLP), Large Language Models(LLM), Retrieval Augmented Generation (RAG)
- Models used: llama3:latest, nomic-embed-text: latest, mxbai-embed-large:latest
- Languages: python3
- Packages and APIs: crewai, faiss-cpu, langchain, chromadb, pypdf, pytest

Link: kavindukalinga/Opensource-LLM-RAG-Chatbot: This repository hosts an NLP project aimed at crafting a chatbot capable of answering questions sourced from provided documents. It leverages open-source large language models (LLM) and retrieval augmented generation (RAG) techniques for this purpose. (github.com)

2.2.3 ISA-Hack 303 Hackathon

Overview:
Hack 303 marked ISA's inaugural Hackathon, held in the final week of March 2024, featuring over 60 participants from various ISA hubs. The event witnessed nine teams competing fervently to tackle real-time complex challenges and innovate AI solutions.

The Hackathon spanned a 30-hour sprint of relentless coding, yielding remarkable products and features now undergoing final refinements for production. Notably, the presence of tech giants such as Google and Microsoft, alongside ISA's in-house subject matter experts, bolstered the teams in devising groundbreaking solutions. A distinguishing feature of the event was the inclusion of six-hour pit stops, where teams engaged in physical activities to earn limited resources for use during the Hackathon.

From honing technical prowess to fostering invaluable networking opportunities, the Hackathon underscored the potential of collaborative innovation. With an emphasis on problem-solving, teamwork, and project development, HACK 303 not only unleashed creativity but also nurtured a culture of agile innovation and teamwork within ISA.

The event concluded with an iftar dinner, symbolizing ISA's commitment to unity and inclusivity as it continues to drive progress in the Tech sphere.

Project:

Participants were assigned the task of developing chatbots using different LLM (Large Language Model) models tailored to address various airline-related tasks. Each project encompassed the following components:

1. Model I/O:

- Prompts: Designing prompts to elicit specific responses from the chatbots.
- Chat models: Implementing conversational models to facilitate interaction with users.
- LLMs (Large Language Models): Utilizing advanced language models for generating coherent responses.
- Output parsers: Developing parsers to extract relevant information from chatbot outputs.

2. Retrieval:

- Document loaders: Loading relevant documents or data sources for chatbot reference.
- Text splitters: Segmenting text inputs into smaller, more manageable components.

- Embedding models: Generating embeddings to represent textual data in a continuous vector space.
- Vector stores: Storing and managing vector representations of text data.
- Retrievers: Implementing algorithms to retrieve relevant information from the stored data.
- Indexing: Organizing and indexing textual data for efficient retrieval.

3. Composition:

- Tools: Developing tools and utilities to aid in chatbot development and deployment.
- Agents: Creating conversational agents to handle specific tasks or user interactions.
- Chains: Constructing chains of conversational components to orchestrate complex interactions and workflows.

Each project aimed to leverage these components to build functional and efficient chatbots tailored to meet the diverse needs of the airline industry. However, due to company policies, specific details regarding the problems addressed and solutions developed during the projects cannot be disclosed.

2.3. Tools Utilized

2.3.1. Flask

Flask is a popular web framework for Python, known for its simplicity, flexibility, and ease of use. It is classified as a micro-framework, meaning it provides the essential tools to build web applications without enforcing a particular project layout or requiring specific libraries beyond what is necessary for routing and templating. This design philosophy offers developers the freedom to choose the components they need and scale their applications accordingly.

Key Features

1**. Simplicity and Minimalism**: Flask follows a minimalistic approach, allowing developers to create a web server with just a few lines of code. Its simplicity does not limit its power; instead, it enables quick development and easy maintenance.

2. **Flexibility: Unlike more opinionated frameworks**, Flask does not impose a particular structure on developers. This flexibility is advantageous for both small applications and large, complex projects.

3. **Extensibility**: Flask's core is simple, but it can be extended through a wide range of Flask extensions. These extensions add functionality such as database integration, form validation, user authentication, and more.

4. **Modularity**: Developers can pick and choose which components and extensions to use, making it possible to keep the application lightweight or enhance it with additional features as needed.

5**. Built-in Development Server and Debugger**: Flask includes a built-in development server and debugger, which streamlines the development process by providing immediate feedback and error tracking.

6. **RESTful Request Dispatching**: Flask supports RESTful request dispatching, making it easy to build REST APIs.

7. **Testing Support**: Flask provides tools for unit testing, making it easier to ensure the quality and reliability of applications.

Common Use Cases

1. Prototyping: Flask is ideal for quickly prototyping ideas and applications due to its minimal setup requirements.

2. APIs: Its support for RESTful request dispatching makes Flask a great choice for developing APIs.

3. Microservices: Flask's lightweight nature and flexibility make it suitable for building microservices in a larger, distributed system.

4. Full-fledged Web Applications: With the right extensions, Flask can be used to build comprehensive web applications, from personal blogs to large-scale enterprise solutions.

Popular Extensions

- Flask-SQLAlchemy: Adds SQLAlchemy support for database operations.
- Flask-WTF: Integrates WTForms for form validation and rendering.
- Flask-Migrate: Handles SQLAlchemy database migrations.
- Flask-Login: Provides user session management for authentication.
- Flask-Restful: Simplifies the creation of REST APIs.

Conclusion

Flask's blend of simplicity, flexibility, and extensibility has made it one of the most popular web frameworks for Python. Its design philosophy empowers developers to build web applications in a way that suits their needs, from lightweight microservices to complex, feature-rich applications. Flask continues to be a go-to choice for Python developers due to its robust ecosystem, active community, and ongoing development.

2.3.2. Docker

Docker is an open-source platform that automates the deployment, scaling, and management of applications using containerization. It allows developers to package an application and its dependencies into a standardized unit called a container, which can run consistently across various environments. Docker revolutionizes the way software is developed, tested, and deployed, making it easier to manage application dependencies and streamline the software development lifecycle.

Key Concepts and Features

1. **Containers**: Containers are lightweight, standalone, and executable packages that include everything needed to run a piece of software, including the code, runtime, libraries, and system tools. They are isolated from the host system and other containers, ensuring consistent behavior across different environments.

2. **Images**: Docker images are read-only templates used to create containers. They are built from a set of instructions written in a Dockerfile and can be stored in a Docker registry, such as Docker Hub.

3. **Dockerfile**: A Dockerfile is a text file that contains a series of instructions on how to build a Docker image. It includes commands for setting up the environment, installing dependencies, copying application code, and specifying the container's startup command.

4. **Docker Engine**: The Docker Engine is the runtime that executes and manages Docker containers. It includes the Docker daemon, a REST API, and a command-line interface (CLI).

5. **Docker Compose**: Docker Compose is a tool for defining and running multi-container Docker applications. It uses a YAML file to configure the application's services, networks, and volumes, making it easy to manage complex applications.

6. **Docker Swarm**: Docker Swarm is Docker's native clustering and orchestration tool. It allows users to manage a cluster of Docker nodes as a single virtual system, providing features like load balancing, scaling, and desired state management.

7. **Docker Hub**: Docker Hub is a cloud-based registry service for sharing and managing Docker images. It hosts a large repository of official and community-contributed images.

Benefits of Using Docker

1. **Portability**: Containers encapsulate all dependencies and configurations, ensuring that applications run consistently across different environments, from development to production.

2. **Isolation**: Containers provide isolated environments for applications, reducing conflicts between software dependencies and improving security.

3. **Scalability**: Docker makes it easy to scale applications horizontally by running multiple instances of containers. Tools like Docker Swarm and Kubernetes can manage container orchestration and scaling.

4. **Efficiency**: Containers are lightweight and share the host system's kernel, making them more resource-efficient compared to traditional virtual machines.

5. **Faster Development and Deployment**: Docker streamlines the development workflow by allowing developers to work in consistent environments, quickly build and test images, and deploy applications rapidly.

Common Use Cases

1. Microservices: Docker is ideal for building and deploying microservices architectures due to its ability to isolate services and manage dependencies efficiently.

2. Continuous Integration/Continuous Deployment (CI/CD)**: Docker facilitates CI/CD pipelines by providing consistent environments for testing and deployment.

3. Development Environments: Developers can use Docker to create reproducible development environments, ensuring consistency across teams.

4. Cloud and Hybrid Deployments: Docker containers can be deployed on various cloud platforms, making it easy to move applications between on-premises and cloud environments.

Conclusion

Docker has transformed the way software is developed, shipped, and run, making it a cornerstone of modern DevOps practices. Its ability to package applications and their dependencies into portable, isolated containers ensures consistent performance across different environments. Docker’s ecosystem, including Docker Hub, Compose, and Swarm, provides a comprehensive toolset for managing the entire application lifecycle, from development to production.

2.3.3.Kubernetes

Kubernetes, often abbreviated as K8s, is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. Originally developed by Google, Kubernetes is now maintained by the Cloud Native Computing Foundation (CNCF). It provides a robust, scalable, and flexible platform for managing distributed applications, enabling organizations to deliver software at a higher velocity and with greater reliability.

Key Concepts and Features

1. **Clusters**: A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. A cluster is managed by the Kubernetes master, which coordinates all activities within the cluster.

2. **Nodes**: Each node in a Kubernetes cluster runs the container runtime (such as Docker), along with an agent called kubelet, and tools for container networking and management.

3. **Pods**: The smallest and simplest Kubernetes object, a pod represents a single instance of a running process in a cluster. Pods can contain one or more containers that share storage, network resources, and a specification for how to run the containers.

4. **Replication Controllers and Replica Sets:** These ensure that a specified number of pod replicas are running at any given time. Replica Sets are the next-generation Replication Controllers and support set-based label selectors.

5. **Deployments**: Deployments provide declarative updates to applications. You describe the desired state of your application, and the Deployment controller changes the actual state to match the desired state at a controlled rate.

6. **Services**: A Kubernetes Service is an abstraction that defines a logical set of pods and a policy by which to access them, typically through a fixed IP address and DNS name. Services decouple the backend pods from clients.

7. **ConfigMaps and Secrets**: ConfigMaps are used to manage configuration data, while Secrets are designed to manage sensitive data like passwords, tokens, and keys.

8. **Namespaces**: Namespaces provide a way to divide cluster resources between multiple users. They are intended for use in environments with many users spread across multiple teams, or projects.

9. **Ingress**: Ingress manages external access to services, typically HTTP, providing load balancing, SSL termination, and name-based virtual hosting.

10. **Persistent Volumes and Persistent Volume Claims**: These provide a way to manage durable storage in a Kubernetes cluster.

Benefits of Using Kubernetes

1. **Scalability**: Kubernetes can scale applications up and down based on demand, ensuring optimal resource utilization.

2. **High Availability**: Kubernetes provides mechanisms for service discovery, load balancing, and self-healing, ensuring applications remain available and resilient to failures.

3. **Portability**: Kubernetes runs on a wide variety of cloud providers and on-premises environments, facilitating hybrid and multi-cloud deployments.

4. **Efficiency**: Kubernetes optimizes resource usage through advanced scheduling techniques, ensuring applications get the necessary resources while maximizing overall cluster efficiency.

5. **Declarative Configuration**: Kubernetes uses declarative configuration files, allowing users to define the desired state of the system and let Kubernetes manage the process of achieving and maintaining that state.

Common Use Cases

1. Microservices Architecture: Kubernetes is ideal for deploying and managing microservices due to its container orchestration capabilities.

2. Continuous Integration and Continuous Deployment (CI/CD): Kubernetes supports automated deployments and rollbacks, making it a key component in CI/CD pipelines.

3. DevOps: Kubernetes promotes a DevOps culture by enabling rapid development, testing, and deployment cycles.

4. Cloud-Native Applications: Kubernetes is designed for cloud-native applications, offering features that facilitate cloud-first development and deployment strategies.

Conclusion

Kubernetes has become the de facto standard for container orchestration, offering a powerful platform for managing modern, cloud-native applications. Its rich feature set, scalability, and portability make it an essential tool for developers and organizations aiming to build and deploy applications efficiently and reliably. By automating the deployment, scaling, and management of containerized applications, Kubernetes helps organizations achieve greater agility, resilience, and operational efficiency in their software development practices.

2.3.4. HELM

Helm is a powerful package manager for Kubernetes, designed to simplify the deployment and management of applications on Kubernetes clusters. Often referred to as the "Kubernetes package manager," Helm enables users to define, install, and upgrade even the most complex Kubernetes applications. Helm streamlines the deployment process by using pre-configured application templates called charts.

Key Concepts and Features

1. **Charts**: A Helm chart is a collection of files that describe a related set of Kubernetes resources. These files include templates and configurations that define the components and settings of an application. Charts can be shared via Helm repositories, making it easy to distribute and deploy applications.

2. **Releases**: A release is an instance of a chart running in a Kubernetes cluster. With Helm, you can deploy multiple releases of the same chart with different configurations.

3. **Repositories**: Helm repositories are collections of charts that are available for download and deployment. The default repository is Helm Hub, but users can add other repositories to access a wider variety of charts.

4. **Templates**: Helm uses Go templates to define Kubernetes manifests. This templating system allows for dynamic configuration, enabling users to customize deployments with parameterized values.

5. **Values**: Values files in Helm provide a way to override the default configurations of a chart. Users can pass their own values files during the installation or upgrade of a chart to tailor it to their specific needs.

6. **Helm CLI**: The Helm Command Line Interface (CLI) is used to interact with the Helm client. It supports a variety of commands for managing charts and releases, such as `helm install`, `helm upgrade`, and `helm rollback`.

Benefits of Using Helm

1. **Simplified Application Management**: Helm abstracts the complexities of Kubernetes resource configurations, allowing users to manage applications with simple commands.

2. **Reusability**: Helm charts can be reused across different environments and projects, promoting consistency and reducing redundancy.

3. **Version Control**: Helm supports versioning of charts, making it easy to manage application updates and rollbacks.

4. **Customization**: Through values files and templating, Helm provides a flexible way to customize deployments without altering the original charts.

5. **Community and Ecosystem**: Helm has a vibrant community and a rich ecosystem of pre-built charts, reducing the time required to deploy and manage applications.

Common Use Cases

1. Application Deployment: Helm is widely used to deploy applications in Kubernetes clusters, simplifying the process and ensuring consistency.

2. CI/CD Pipelines: Helm integrates well with CI/CD tools, enabling automated deployments and updates in development workflows.

3. Configuration Management: Helm helps manage complex configurations through its templating and values systems, making it easier to deploy applications in different environments.

4. Microservices: Helm’s ability to manage multiple Kubernetes resources as a single unit makes it ideal for deploying and managing microservices architectures.

Conclusion

Helm significantly enhances the Kubernetes experience by providing a powerful and flexible way to manage applications. Its templating and packaging capabilities streamline deployment processes, making it easier for developers and operators to work with Kubernetes. Helm’s extensive ecosystem of pre-built charts, combined with its ability to customize and manage applications efficiently, makes it an essential tool in modern DevOps practices and cloud-native application management.

2.3.5. Oracle Cloud Infrastructure

Oracle Cloud Infrastructure (OCI) is Oracle Corporation's comprehensive suite of cloud services, designed to enable businesses to run their workloads with high performance, reliability, and security. OCI provides a range of cloud solutions including computing, storage, networking, database, and various platform services, catering to a variety of enterprise needs. It is built to support mission-critical applications and workloads, offering a robust infrastructure for modern cloud-native and traditional on-premises applications.

Key Features and Services

1. **Compute Services**: OCI provides a variety of compute options, including virtual machines, bare metal servers, and high-performance computing (HPC) clusters. These services offer scalability and flexibility for running diverse workloads, from small applications to large-scale enterprise systems.

2. **Storage Services**: OCI's storage offerings include block storage, object storage, file storage, and archive storage. These services ensure data durability, availability, and security, meeting the needs of applications ranging from databases to big data analytics.

3. **Networking**: OCI offers a robust networking infrastructure with Virtual Cloud Networks (VCNs), dynamic routing, load balancing, and VPN connectivity. This allows for secure and scalable network architectures, supporting both hybrid cloud and multi-cloud deployments.

4. **Database Services**: Oracle's database services on OCI include Oracle Autonomous Database, Oracle Exadata Cloud Service, and Oracle Database Cloud Service. These services provide high performance, scalability, and automation, simplifying database management and operations.

5. **Security**: OCI emphasizes security with features like identity and access management (IAM), data encryption, security zones, and comprehensive monitoring. These features help protect workloads and ensure compliance with industry standards.

6. **Developer Tools**: OCI supports a range of developer tools and services, including Oracle Container Engine for Kubernetes (OKE), Oracle Functions (serverless computing), and DevOps services for continuous integration and deployment (CI/CD).

7. **Analytics and AI**: OCI provides advanced analytics and AI services, including Oracle Analytics Cloud, Oracle Data Science, and Oracle Big Data Service. These services enable businesses to derive insights from their data and build intelligent applications.

8. **Integration Services**: OCI offers integration tools like Oracle Integration Cloud, which helps connect applications and automate workflows across on-premises and cloud environments.

Benefits of Using OCI

1. **Performance and Reliability**: OCI is designed for high performance and reliability, offering service-level agreements (SLAs) for availability, manageability, and performance.

2. **Cost-Effectiveness**: OCI provides competitive pricing and cost management tools, allowing businesses to optimize their cloud spending and achieve cost savings.

3. **Flexibility and Scalability**: OCI's broad range of services and scalable infrastructure supports diverse workloads and business needs, enabling easy scaling of resources as demand grows.

4. **Security and Compliance**: OCI incorporates comprehensive security measures and compliance certifications, ensuring data protection and regulatory compliance.

5. **Integration with Oracle Applications**: OCI offers seamless integration with Oracle's extensive suite of enterprise applications, making it an ideal choice for businesses already using Oracle software.

Common Use Cases

1. Enterprise Applications: Running Oracle E-Business Suite, JD Edwards, PeopleSoft, and other Oracle applications on OCI for enhanced performance and reliability.

2. Database Management: Utilizing Oracle Autonomous Database and Oracle Exadata Cloud Service for high-performance, scalable, and automated database management.

3. Hybrid and Multi-Cloud Strategies: Implementing hybrid cloud solutions by integrating OCI with on-premises infrastructure and other cloud providers.

4. Big Data and Analytics: Leveraging OCI's analytics and AI services to process and analyze large datasets, gaining valuable business insights.

5. Cloud-Native Applications: Developing and deploying cloud-native applications using OCI's developer tools, Kubernetes services, and serverless computing.

Conclusion

Oracle Cloud Infrastructure stands out as a robust and comprehensive cloud platform tailored for enterprise needs. Its wide array of services, emphasis on performance and security, and seamless integration with Oracle's enterprise applications make it a compelling choice for businesses looking to modernize their IT infrastructure and leverage the benefits of cloud computing. Whether for running traditional enterprise workloads or developing modern cloud-native applications, OCI provides the necessary tools and capabilities to support a broad spectrum of use cases and industries.

2.4. Non-Technical Experience Gained at ISA

2.4.1.Professional Practices

Professional practices are the backbone of ethical conduct, adherence to best practices, and social responsibility within the engineering field. My internship experience as a DevOps Engineer at ISA provided invaluable insights into five pivotal skills highly esteemed in the industry.

1. Communication: Effective communication is paramount in computer engineering. It facilitates the articulation of ideas, seamless sharing of concepts, and efficient proposal of solutions. Through clear communication with peers, management, and clients, collaboration is enhanced, leading to more effective problem-solving and increased project success rates.

2. Transparent Issue Discussion: Openly discussing concerns or issues with supervisors fosters a harmonious work environment conducive to problem resolution and diverse perspectives. Constructive and respectful dialogue allows for the exchange of viewpoints, fostering better working relationships and more effective problem-solving strategies.

3. Embracing Feedback for Growth: Feedback serves as a catalyst for personal and professional development. It offers valuable insights into progress, strengths, and areas for improvement. By embracing feedback, individuals can enhance their skills, foster collaboration, and cultivate strong workplace relationships. Viewing feedback as a tool for character development fuels continuous improvement and drives excellence.

4. Emphasis on Teamwork: Teamwork is indispensable in the software industry, enabling collaboration, innovation, and effective knowledge sharing. Through teamwork, developers can collectively brainstorm ideas, overcome challenges, and devise innovative solutions. This collaborative environment boosts productivity, communication, and creativity, ultimately leading to the development of superior software products.

5. Adaptability to Workplace Culture: Adapting swiftly to workplace culture fosters unity and understanding among team members. Embracing the culture facilitates effective communication and provides a platform for expressing ideas and opinions, reducing stress and conflicts. Adaptability allows individuals to navigate diverse work environments seamlessly, ensuring they can thrive regardless of circumstances.

During my internship, I recognized the paramount importance of these professional practices and their profound impact on personal and professional growth. By embracing these skills, I not only honed my technical expertise but also cultivated the ability to collaborate effectively, communicate confidently, and adapt seamlessly to diverse work environments. I am confident that these skills will continue to guide me towards success in the engineering field, enabling me to make meaningful contributions to future projects while upholding the highest standards of professional practices in the industry.

3. Conclusion

3.1. Training Establishment's ability to provide useful Industrial Training

ISA, with its specialized focus on providing IT consultancy services to the aviation industry, stands out for its ability to offer comprehensive and valuable industrial training. The company's commitment to understanding the unique challenges and requirements of the aviation sector ensures that its programs are tailored to meet the specific needs of both the industry and the individual trainees.

One of ISA's key strengths lies in its team of experts who possess extensive industry knowledge and expertise, enabling the delivery of highly effective and customized solutions. This expertise ensures that trainees gain practical insights and skills that are directly applicable to their roles within the aviation sector.

Furthermore, ISA's supportive and positive work culture plays a significant role in enhancing the effectiveness of its training programs. The company fosters a strong work ethic and encourages employees to embrace challenges, creating an environment conducive to learning and growth. This culture extends to the training programs, where supervisors provide close guidance and personalized task assignments based on the trainees' skills, weaknesses, and interests, facilitating personal growth and contributing to the company's overall development.

ISA's commitment to diversity and inclusion further enhances the value of its programs. With a team comprised of individuals from diverse backgrounds, ISA offers a rich tapestry of ideas and perspectives, enriching the learning experience and exposing trainees to a wide range of viewpoints and approaches. Additionally, ISA's inclusive culture ensures that every trainee feels valued and supported throughout their journey.

As ISA continues to grow, it remains dedicated to providing opportunities for professional development and advancement. This commitment ensures that trainees not only receive valuable training but also have the opportunity to build fulfilling careers within the company. By joining ISA's dynamic team, trainees become part of a collaborative environment that encourages creativity, values differences, and embraces challenges.

3.2. Training personnel of the Establishment

Under the guidance of Dilan Anuruddha, the Head of Software Engineering at the company, and Rimaz Fawzer, the leader of the DevOps team, I experienced an internship filled with invaluable learning opportunities and growth. Dilan Anuruddha and Rimaz Fawzer demonstrated remarkable patience throughout my journey, consistently encouraging me to tackle challenges independently during daily tasks.

One of the most admirable qualities of them was their approach to mentorship—they never imposed unnecessary pressure but instead allowed me to learn from my mistakes and complete tasks autonomously. Their close monitoring of my progress ensured that I had a solid understanding of the tasks at hand, and whenever I lacked background knowledge, they supported my self-learning efforts and trusted me with complex assignments. This trust empowered me to gain valuable experience and knowledge, contributing significantly to my professional development.

I also found the team environment, comprising Ravindra Singh and Manoj Nilanga, who assigned tasks and taught me everything, to be incredibly supportive and encouraging. They consistently motivated me to expand my technical knowledge and were always willing to lend a helping hand, even on busy days and occasionally after working hours. Their positivity and willingness to nurture talent greatly contributed to my growth and development during my internship at the company.

Moreover, they valued not only my technical skills but also my soft skills. They provided opportunities for me to collaborate with new team members, fostering growth in various areas beyond technical expertise. This holistic approach to mentorship enhanced my overall skill set and prepared me for future challenges in the industry.

Overall, my experience was instrumental in shaping my professional journey and preparing me for the challenges ahead in the field of software engineering.

3.3. Deficiencies in the Training Establishment and possible suggestions

ISA has successfully cultivated an inclusive and collaborative work environment, which has been highly beneficial for interns. However, to further foster camaraderie and team cohesion among interns, the company could arrange additional social gatherings. These events would facilitate more informal interactions, allowing interns to forge lasting connections. Given the virtual nature of the internship, ISA could explore both in-person gatherings and virtual meet-ups to accommodate varying preferences. These gatherings would offer valuable networking opportunities for interns to connect with senior staff members, fostering meaningful relationships within the organization.

Furthermore, ISA could enhance the internship experience by organizing webinars, tech talks, or workshops tailored specifically for interns. These sessions would deepen interns' understanding of relevant topics and provide a platform for engagement with industry experts. By exposing interns to valuable insights and trends, ISA can enrich their professional development and offer practical knowledge that enhances their overall internship experience.

Implementing these initiatives would elevate ISA’s internship program, providing a more enriching and rewarding experience for interns. The inclusion of social events and educational opportunities would contribute to a stronger sense of community within the company and empower interns to excel in their professional growth while establishing valuable connections in the industry.

3.4. Overall Industrial Training programme organised by the University and NAITA

The Department of Electronic and Telecommunication Engineering has demonstrated exceptional commitment to assisting students in securing high-quality and valuable training opportunities. The guidance provided, including assistance with crafting CVs, the CV review program, and conducting mock interviews by faculty members, has proven immensely beneficial in navigating the competitive industry landscape and securing suitable internships. Throughout the internship period, the faculty remained actively engaged, providing ongoing support and monitoring the training establishment to help students overcome challenges, thereby ensuring a successful internship experience.

Drawing from my personal experience, I advocate for internships to be scheduled in the final semester, specifically in the second semester of the fourth year. This arrangement enables students to complete their internship and seamlessly transition into full-time employment or explore further opportunities in the industry. Returning to university after gaining industry experience can disrupt the continuity of learning and professional development.

3.5. Overall experience

During my internship tenure at ISA, spanning from December 4th, 2023, to May 21st, 2024, I had the privilege of engaging in various projects across multiple domains within the DevOps team. These projects encompassed diverse areas such as Prometheus alert enrichment, PostgreSQL database encryption, LLM RAG chatbot, and more. This immersive experience provided me with invaluable hands-on exposure to cutting-edge technologies like Kubernetes and Cloud Technologies. While my primary focus revolved around DevOps tasks, I seized the opportunity to contribute to a myriad of development-related projects, thereby gaining a comprehensive understanding of software development.

Additionally, I participated in the company-organized hackathon, ISA-hack-303, and progressed to stage 2, which was held in Sharjah. This experience afforded me the opportunity to visit another country during my internship and engage in collaboration with industry giants such as Microsoft, OpenAI, and others. It was an enriching experience to compete alongside and gain insights from these tech pioneers.

From the onset of my internship, I committed myself to mastering the fundamentals of the projects and the technologies employed. This foundational knowledge empowered me to make meaningful contributions from the outset. Beyond merely delivering functional code, I prioritized optimizing efficiency, adhering to coding standards, and ensuring code maintainability for future developers.

Throughout my internship journey, I encountered challenges such as debugging and problem-solving. However, the unwavering support from my supervisor and teammates proved instrumental. Their mentorship and guidance bolstered my confidence, enabling me to become proficient in resolving issues and eventually supporting and mentoring others.

Collaborating with seasoned professionals at ISA, including software engineers, senior software engineers, and technical leads, significantly enriched my learning experience. Interacting with fellow interns from diverse academic backgrounds broadened my perspective and fostered a strong sense of camaraderie. This collaborative environment underscored the importance of effective communication and maintaining a professional yet amicable demeanor.

Reflecting on my time at ISA, I take pride in the progress I've made. The internship not only enhanced my technical skills but also honed my ability to learn quickly, adapt to challenges, manage my workload efficiently, and tackle complex problems effectively.

I wholeheartedly recommend ISA as an exceptional destination for future interns. The company's intellectually stimulating environment, exciting projects, and support from experienced professionals create a rewarding and growth-oriented internship experience. My tenure at ISA has equipped me with the confidence and capabilities to approach future endeavors with enthusiasm and a growth mindset.

References

[1] “Information Systems Associates” isa.ae

[2] ACCELaero - Accelerate the growth of your Airline

[3] kavindukalinga (Chandrasiri Y U K K) (github.com)

-