Project ID: 438037

Data Ingestion

AN INTERNSHIP REPORT

Submitted By

Fazal Mansuri
200280116012

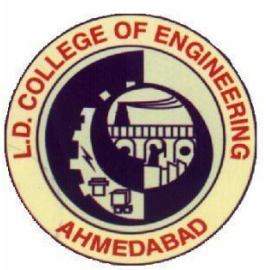
In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

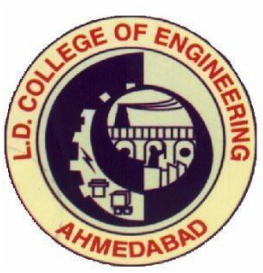
Information Technology

L.D. College of Engineering, Navrangpura,
Ahmedabad - 380015



Gujarat Technological University, Ahmedabad

[April, 2024]



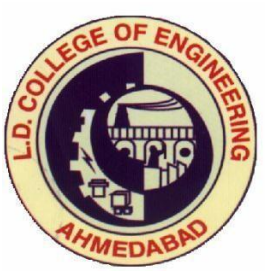
L.D. COLLEGE OF ENGINEERING
Navrangpura, Ahmedabad, 380015

CERTIFICATE

This is to certify that the internship report submitted along with the project entitled **Data Ingestion** has been carried out by **Fazal Mansuri (Enrollment No. 200280116012)** under my guidance in partial fulfilment for the degree of Bachelor of Engineering in Information Technology, **8th Semester** of Gujarat Technological University, Ahmedabad during the academic year 2023-24.

Dr. Hiteishi Diwanji
Internal Guide

Dr. Hiteishi Diwanji
Head of the department,
Information Technology



L.D. COLLEGE OF ENGINEERING
Navrangpura, Ahmedabad, 380015

DECLARATION

I hereby declare that the Project report submitted along with the Internship project entitled **Data Ingestion** submitted in partial fulfilment for the degree of Bachelor of Engineering in **Information Technology** to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me at **Crest Data Systems** under the supervision of Internal Guide **Dr. Hiteishi Diwanji** and External Guide **Rahul Virpara** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Name of the Student
Fazal Mansuri

Sign of Student

ACKNOWLEDGEMENT

The satisfactory completion of the internship duration would not have been without the people who guided and supported me with my journey as an intern at Crest Data Systems. Firstly I would like to thank Mrs Devanshi Raichura who is Head of HR department at Crest Data Systems, who provided me this great opportunity to do an internship with the organization. I would also like to thank Mr. Rahul Virpara who was my mentor at the company during my internship tenure.

I would especially like to express my gratitude towards Prof.Hiteishi Diwanji of LD college of Engineering. I am thankful to her for her continuous support and encouragement. She guided me whenever i faced difficulties.

I would also like to express my gratitude towards Jainil Desai for guiding me with his experience.

Fazal Mansuri

200280116012

ABSTRACT

This report is about my experience of an internship at Crest Data Systems. This internship taught me many things regarding the software industry and security tools. I learnt about the concept of Go lang and use cases of go lang in a project. After learnt about apache kafka and implementation of kafka in projects that build understanding of messaging queue and event streaming.

After completion of Go lang training, I worked with data ingestion project that taught me shell script, powershell script, lua script, syslog, pipeline and also cleared some concepts of microservices.

During this tenure I got exposure to the industrial environment and practices which I feel is the primary objective of internships. I feel these learnings will help me to grow in my professional life.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABREVATIONS

AWS	Amazon Web Services
SQS	Simple Queue Service
CI/CD	Continuous Integration / Continuous Deployment
SIEM	Security Information and Event Management
SOAR	Security Orchestration, Automation, and Response
SDK	Software Development Kit
API	Application Programming Interface
IDE	Integrated Development Environment

TABLE OF CONTENTS

CHAPTER 1: OVERVIEW OF THE COMPANY

1.1 HISTORY

Fig 1.1 Company Logo

Crest Data Systems is a leading provider of custom solutions in the areas of Data Analytics, Cyber Security, ITOps/AIOps, DevOps, and Cloud based in the Bay Area, California founded in 2013. Crest is a fast-growing company working on bleeding-edge technologies with a broad customer-base that includes several Fortune 500 corporations as well as some of the hottest Silicon Valley Start-ups. With an end-to-end services portfolio that includes Development, Consulting/Professional, and 24x7 Managed services, crest helps customers build industry-leading solutions that help them outperform competition and stay ahead of the innovation curve.

1.2 DIFFERENT PRODUCTS/SCOPE OF WORK

Different Products

- Datadog Apps
- ElasticSearch to Splunk Migration
- Managed cloud exchange
- OpenStack Analytics

Scope of work

Crest Data Systems (CDS) specializes in providing a wide range of services including software development, product integrations, cloud operations, and managed services to both global enterprises and innovative start-ups.

- **Data Analytics**

1) Security Data Analytics

Crest Data Systems (CDS) leverages Security Information and Event Management (SIEM) technology, tools, and platforms to assist enterprises and smaller organizations in enhancing their security posture. Through security analytics, which involves data collection, aggregation, and analytics, CDS aids in security monitoring, malware research, and threat detection. By collaborating with leading technology vendors such as Splunk, HP ArcSight, IBM QRadar, and ElasticSearch, CDS designs and delivers robust SIEM solutions to enable proactive alerting for attempted attacks or ongoing incidents.

2) Data Analytics Integrations

Recognized as a market leader in Data Analytics integrations, Crest Data Systems (CDS) has accomplished the delivery of over 2000 integrations, facilitating profound business insights for our clients. Our adept professionals craft integrations by harnessing data from syslog, APIs, or custom scripts, ensuring comprehensive data collection. This meticulous approach allows us to correlate data effectively, providing valuable business insights to our clientele.

- **Security**

1) Security Tools and Platform Integrations

As one of the largest Enterprise Security integration developers, CDS can help extend capabilities of your organization's security products in the areas of application security, network security, cloud security, endpoint security, threat Intelligence, incident management, and Identity & Access management. Some of our popular integrations are SIEM (Splunk Enterprise Security, IBM QRadar, MicroFocus ArcSight, Google Chronicle, etc.), SOAR (Splunk Phantom, Palo Alto Demisto, ServiceNow SecOps, etc.), Firewalls (Cisco, Palo Alto, Checkpoint, etc.) and EDR (Crowdstrike, Symantec, SentinelOne, McAfee, Carbon Black, etc.).'

2) Managed Security Orchestration, Automation, and Response (SOAR) Services

We accelerate and automate security operations and incident resolution by orchestration of security resources and integrating disparate security systems on SOAR Platforms.

- **DevOps**

DevOps spans across the entire delivery pipeline to improve deployment frequency, lower failure rate of new releases, shorten lead time between fixes, and enable faster mean time to recovery. Simple processes become increasingly programmable, automated, and dynamic by using a DevOps approach.

- **CloudOps**

At Crest Data Systems, our CloudOps team specializes in simplifying the process of moving your infrastructure, data, platforms, and applications to the cloud. Whether you're looking to modernize your IT infrastructure, optimize data management, or streamline application deployment.

1.3 ORGANIZATION CHART

Malhar Shah (Co-founder , CEO)
Neha Shah (Co-founder , CTO)
Nirav Thakkar (CFO)
Clowin Fernandes (VP , Engineering)
Vance Cochrane (Director , Business developmant)
Anant Shah (Director , Cloud Operations)
Kushal Shah (Director , IT)
Devanshi Raichura (Director , HR)

1.4 CAPACITY OF ORGANIZATION

The capacity of organization is 1000 – 5000 employees.

CHAPTER 2: OVERVIEW OF DIFFERENT PROCESS BEING CARRIED OUT IN THE COMPANY

2.1 DIFFERENT DEPARTMENTS

Business Department

The Business department at Crest Data Systems (CDS) specializes in conducting thorough market analysis and requirement assessments tailored to the services offered by our organization. With a dedicated focus on understanding market trends, customer needs, and industry dynamics, our team ensures that our services align seamlessly with the evolving demands of our clients. By staying attuned to market shifts and customer preferences, we continuously refine our strategies to deliver innovative solutions that address emerging challenges and opportunities in the IT landscape

HR Department

At Crest Data Systems (CDS), the HR department takes charge of organizing various activities to foster a vibrant and engaging work culture. In addition to orchestrating events aimed at employee well-being and team building, the HR department also oversees the entire hiring process and manages salary administration for our workforce. By facilitating a seamless recruitment experience and implementing fair and competitive salary structures, our HR team ensures that we attract and retain top talent, contributing to the company's overall success and growth.

QA Department

Various departments at Crest Data Systems (CDS) are actively involved in conducting diverse testing processes on software products. This includes platform testing, system testing, and other specialized testing methodologies. Through rigorous testing protocols, our teams ensure the reliability, functionality, and performance of the software solutions we deliver. Whether it's testing on specific platforms or evaluating the system as a whole, each department collaborates to uphold the highest standards of quality assurance, thereby ensuring the seamless deployment and operation of our software offerings.

SRE Department

Crest Data Systems' Site Reliability Engineering (SRE) department is responsible for executing tasks related to observability and managing cloud migrations. With a focus on enhancing system reliability and performance, the SRE team implements advanced monitoring and observability practices to ensure seamless operation of our systems and services. Additionally, they oversee the smooth transition of infrastructure and applications to cloud environments, utilizing their expertise to optimize performance, scalability, and resilience. Through proactive management and innovative solutions, the SRE department plays a critical role in maintaining the stability and efficiency of our technology ecosystem.

Splunk Department

The Splunk Operations department at Crest Data Systems (CDS) specializes in managing all aspects related to Splunk, including Splunk Enterprise and Splunk Security. This dedicated team ensures the smooth operation, optimization, and customization of Splunk deployments to meet the specific needs of our clients. From configuring Splunk instances to analyzing data and generating actionable insights, our experts leverage their deep understanding of Splunk technologies to maximize its capabilities and enhance security, monitoring, and data analytics functionalities. By providing comprehensive support and maintenance services, the Splunk Operations team plays a crucial role in empowering organizations to harness the full potential of Splunk for their business operations

DevOps Department

The DevOps Department at Crest Data Systems oversees all production-related operations within the organization. This dynamic team is responsible for streamlining the development, deployment, and maintenance processes to ensure efficient and reliable delivery of software products and services. Leveraging cutting-edge DevOps practices and tools, they automate workflows, optimize infrastructure, and implement continuous integration and continuous deployment (CI/CD) pipelines. By fostering collaboration between development and operations teams, the DevOps Department accelerates innovation, improves time-to-market, and enhances overall system stability and performance

2.2 LIST THE TECHNICAL SPECIFICATIONS OF MAJOR EQUIPMENT USED IN EACH DEPARTMENT

Processor : Intel core i5-11800H

Display : 15 Inch FHD 250nits

Memory : 8 GB

Storage : 1TB SSD

Graphics : NVIDIA 2GB

OS : Ubuntu, RHEL

2.3 SCHEMATIC LAYOUT OF END PROJECT

Here is the process diagram shown about the process that is being followed for the development

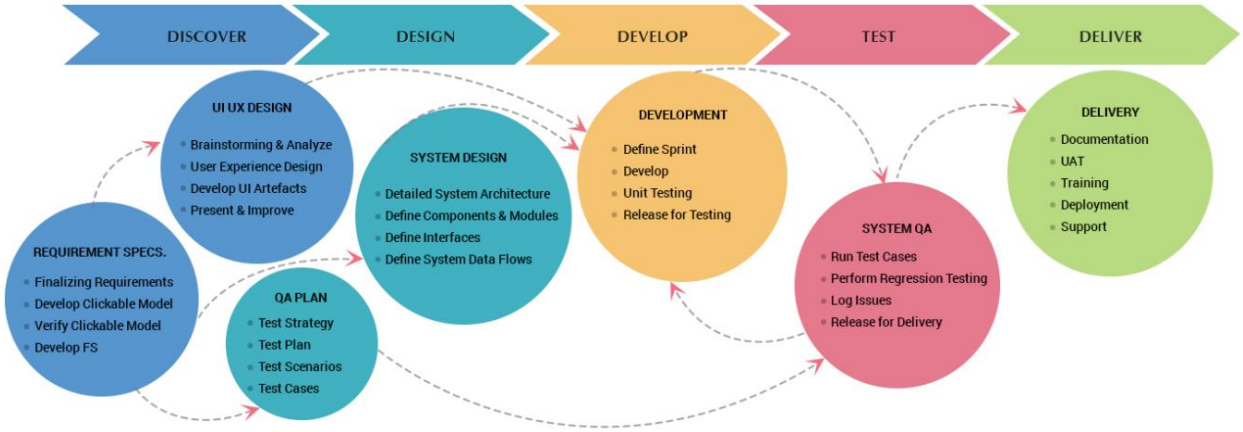


Fig 2.1 Production Process

2.4 EXPLAIN IN DETAILS ABOUT EACH STAGE OF PRODUCTION

- **DISCOVER**
  Brainstorming & Analyze: This involves identifying the problem or opportunity for the software product, as well as researching the target market and competition. User experience (UX) design is also considered at this stage.
  Develop UI Artefacts: This involves creating mockups and prototypes of the user interface (UI) to get user feedback and ensure that the product is user-friendly.
  Present & Improve: This involves presenting the brainstorming and UI artefacts to stakeholders and getting their feedback. The team will then improve the product based on this feedback.

- **DESIGN**
  UI/UX Design: This stage involves defining the look and feel of the software product, as well as how users will interact with it. This includes creating wireframes, mockups, and prototypes.
  System Design: This stage involves defining the architecture of the software product, as well as how it will be built. This includes defining components, modules, interfaces, and data flows.

- **DEVELOP**
  Define Sprint: This involves breaking down the development process into smaller, more manageable pieces of work called sprints.
  Develop: This involves writing the code for the software product. Unit testing is also conducted at this stage to ensure that individual units of code are working correctly.

- **TEST**
  UAT Documentation: This involves creating user acceptance testing (UAT) documentation, which is used to test the software product from the end user's perspective.
  System QA: This involves performing system testing to ensure that the software product meets all of the requirements and that it is free of bugs.
  Develop Test Cases: This involves creating test cases, which are specific steps that can be used to test the software product.

- **DELIVER**
  Release for Delivery: This involves packaging the software product and making it ready for delivery to the customer.
  Deployment: This involves deploying the software product to the customer's environment.
  Training: This involves training the customer on how to use the software product.
  Support: This involves providing ongoing support to the customer after the software product has been delivered.

CHAPTER 3: INTRODUCTION TO INTERNSHIP

3.1 INTERNSHIP SUMMARY

The main objective of the inclusion of internship into curriculum is to make a student industry ready and make him learn how to apply his/her knowledge to encounter industrial challenges. My experience of the internship at Crest Data Systems was a great learning experience. Initially i got training of different tools like git, linux, vscode, jira,etc then Go language training was provided.

I created a training project that collects data from apache kafka through consumers and performs processing on it as well as stores it into a database.

After completion of Go lang training my internship project data ingestion started.

3.2 PURPOSE

The purpose of the internship is to make a student industry ready and be able to make a career in the ever growing industry. Academic knowledge makes students aware about the technologies and terminologies, yet the market demands advanced skill sets and technologies. The purpose is to expose students to industrial knowledge as well as purpose of internship was to gain practical experience in developing web applications using the Go lang, work on real-world projects, and collaborate with a team of experienced developers. The internship helped me to develop my skills in backend development.

3.3 OBJECTIVE

- Provide hands-on experience in building cross-platform web applications using Go language.
- Offer the opportunity to work on real-world projects and gain practical experience.
- Develop skills in Backend development and work closely with experienced developers.
- Receive mentorship and guidance to prepare for a career in Backend development.
- Access to the latest tools and technologies in the industry.

3.4 SCOPE

The project encompasses the development of the On-Prem Collector and Receiver system supporting various data types, initially focusing on syslog. The system should ensure seamless integration with other tools, high reliability with monitoring features, and efficient data retention capabilities.

3.5 TECHNOLOGY AND LITERATURE REVIEW

During this Internship I came across various technologies which were necessary building blocks for web based application and software.

- **Git**

  Git is a famous free and open source distributed version control system. Git is used to manage various versions of the software. I learnt about branching using git and also learnt about various commands of git like commit, push, cherry-pick, reset, etc which are essential in efficient version control and collaboration in software development.

Fig 3.1-Git Logo

- **Jira**

  Jira is a project management tool developed by Atlassian. It helps teams to plan, track and manage their work efficiently.

  During jira training i learnt how to raise issue (ticket) and how to assign tasks through tickets.

Fig 3.2 Jira Logo

- **Linux**

  In Linux training we have learnt about various concepts of operating system and commands of linux (like pwd, ls, chmod,ps,etc) along with overview of linux architecture, linux file structure and basics of shell scripting.

Fig 3.3-Linux Logo

- **Go lang**
  Go lang training is conducted into two parts, first is golang training and second part weekly project using the golang concepts.

  Topics covered in Go lang training :
  - Brief introduction to the Go programming language.
  - Explanation of variables and data types in Go.
  - Overview of control flow constructs and functions in Go.
  - Explanation of importing packages and working with arrays.
  - Overview of slices, maps, and pointers in Go.
  - Explanation of structs, defer statement, and panic/recover mechanism.
  - Overview of interfaces, goroutines, and concurrency handling in Go.
  - Explanation of waitgroups, context package, and mutexes for synchronization.
  - Http server in Go lang

  Weekly project :
  Create a robust system using Go lang that collects data from apache kafka and perform processing and store data into the database. This project implementation helps me to use concepts like channel, goroutines, context, waitgroups, producer-consumer, graceful shutdown in development.

3.6 INTERNSHIP PLANNING

3.6.1 Development Approach

Software Project planning starts before technical work start.

The image below depicts the project planning steps and describes the Software Development Life Cycle.

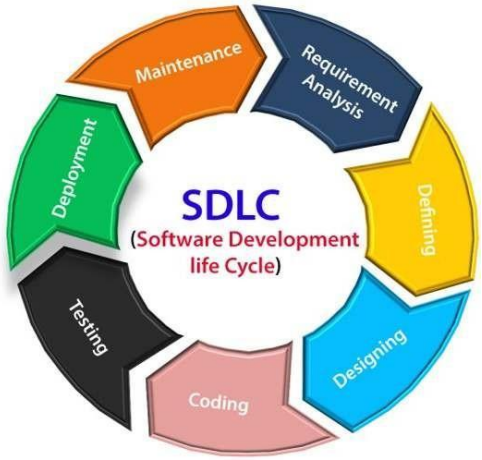


Fig 3.4 SDLC

3.6.2 Role and Responsibilities

- Investigation
- Requirement Analysis
- Design
- Coding
- Testing

3.6.3 Group Dependency

- Individual Dependency

3.7 INTERNSHIP SCHEDULING

3.7.1 Internship Development Approach and Justification

The project follows **"Agile methodology"**. Agile methodology is a popular approach to software development that emphasizes flexibility, collaboration, and iterative progress.

- **Sprint Planning**: During this phase, identifies the goals and objectives for the upcoming sprint. And will review the user stories and prioritize the tasks based on importance and complexity.

- **Sprint Execution:** During this phase, I will work on developing the product Architecture. The team will use Agile practices such as daily stand-ups, sprint demos, and retrospectives to ensure that the project is on track.
- **Continuous Integration/Continuous Deployment (CI/CD):** In this phase, I will implement CI/CD pipelines to automate the deployment of the infrastructure. I will use tool like Github Actions to automate the process of testing, building, and deploying the infrastructure.
- **Sprint Review:** In this phase, I will review the progress made during the sprint and ensure that the objectives have been met. I will also demo the project architecture and get feedback from stakeholders.
- **Sprint Retrospective:** During this phase, I will reflect on the sprint and identify areas for improvement. I will discuss what worked well and what could be improved for the next sprint.

3.7.2 Internship Scheduling

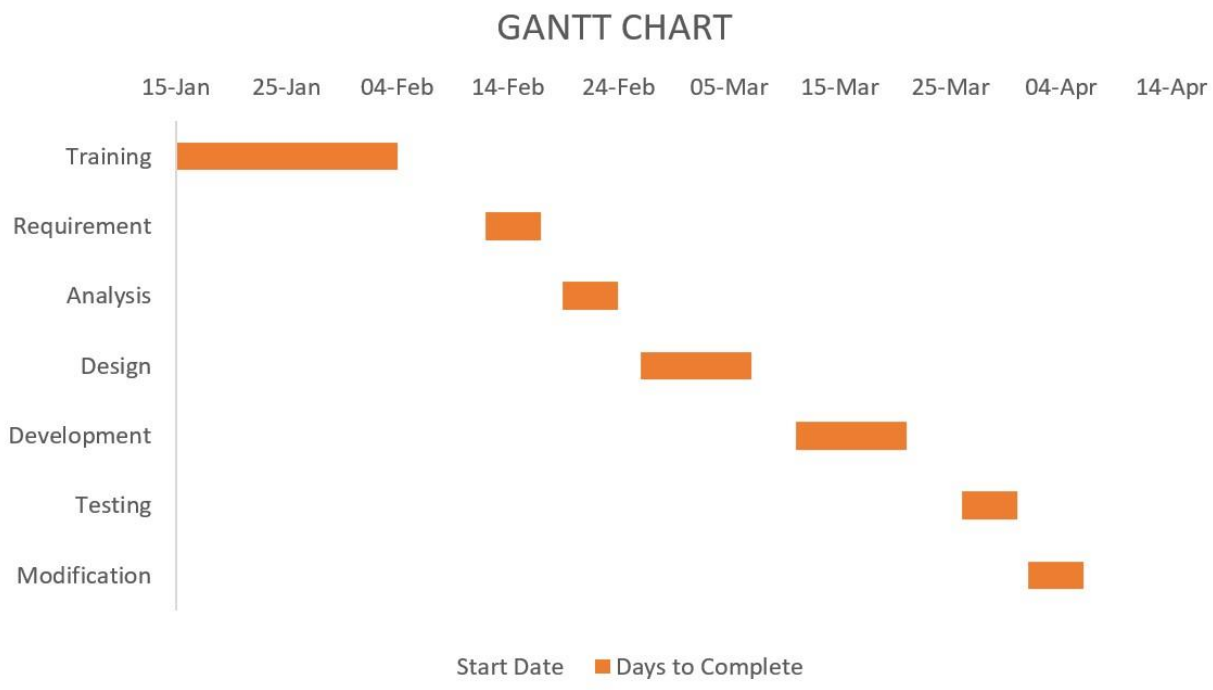


Fig 3.5 Gantt Chart

CHAPTER 4: SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM

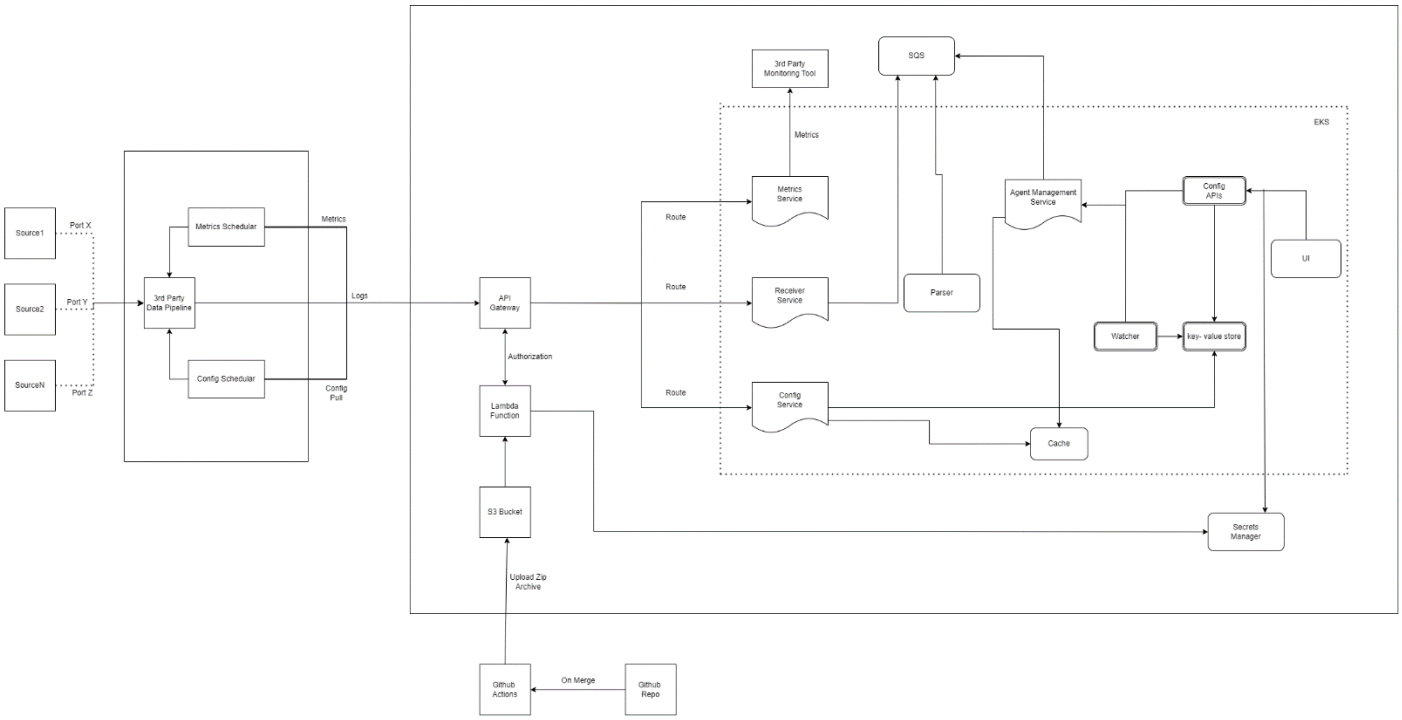


Fig 4.1 Architecture Diagram

4.2 ACTIVITY DIAGRAM

4.2.1 Activity Diagram for Installing Data Collector

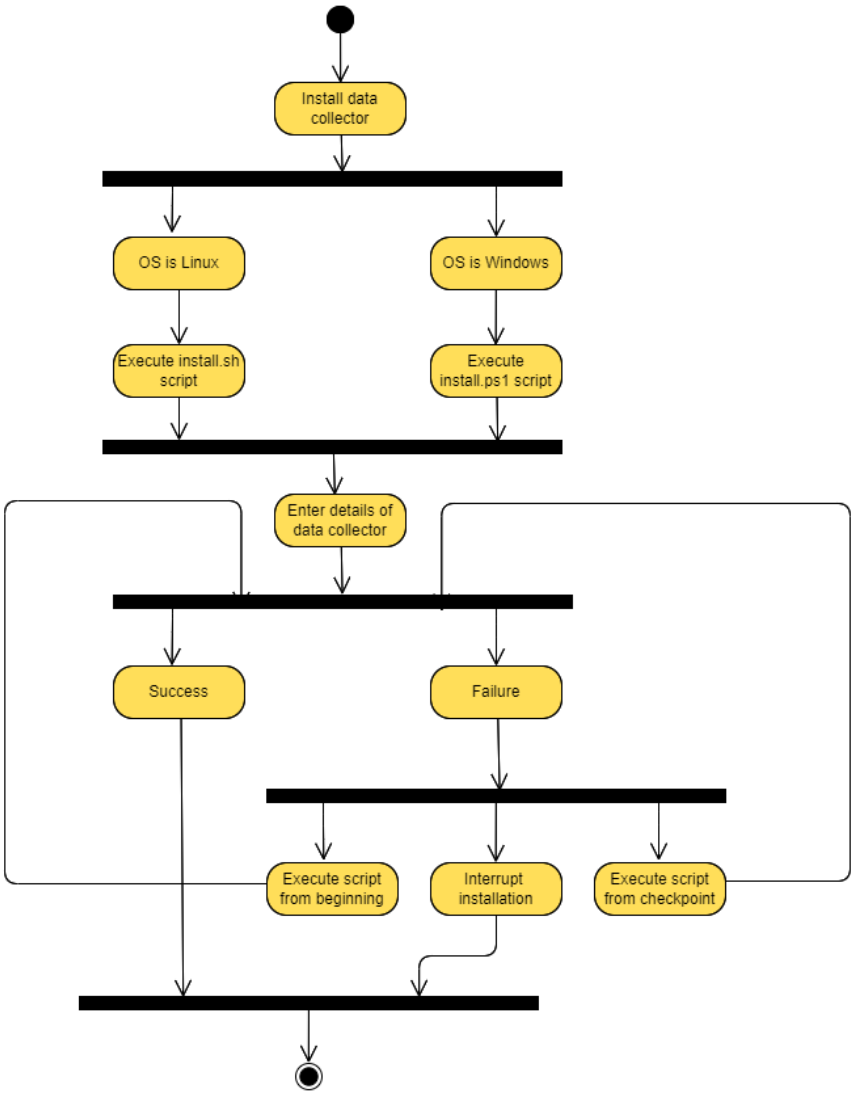


Fig 4.2 Activity Diagram for Installing Data Collector

Description:
Above activity diagram represents the process of installing the data collector to the system. To initiate the collection of logs from the device, users must install the data collector by executing a script. The installation process supports two primary operating systems: Linux and Windows. If any failure arises during execution, users have the option to either resume installation from a designated checkpoint or restart the entire process. If the installation encounters any interruptions, it halts, necessitating users to begin the installation anew.

4.2.2 Activity diagram for configure data source and attach with data collector

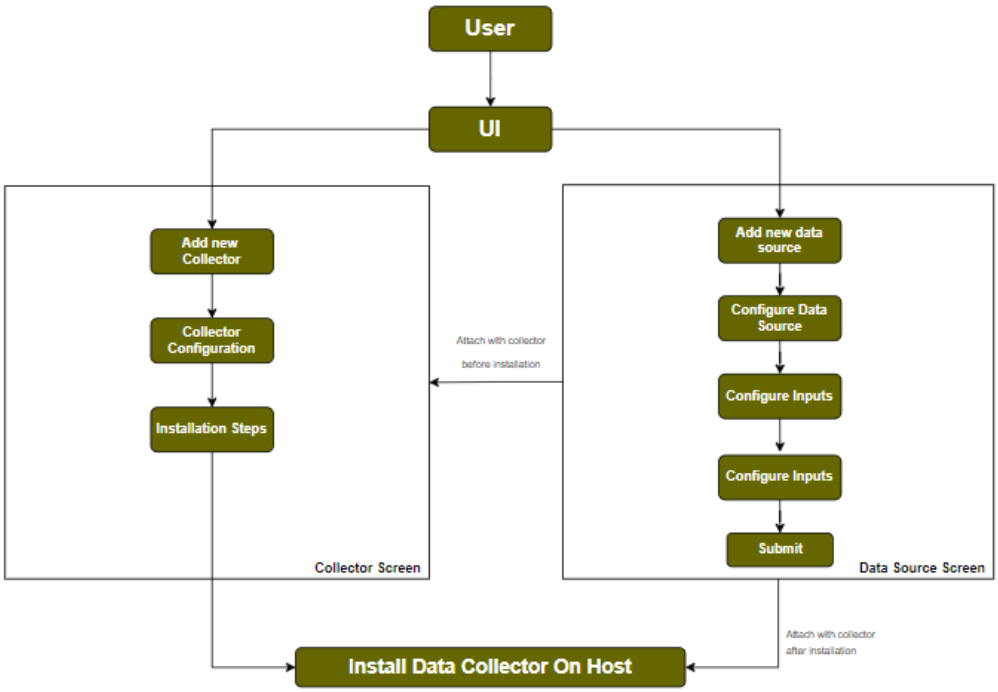


Fig 4.3 Activity Diagram for Configure Data Source and Attach with Data Collector

Description:

1. The above activity diagram illustrates the sequential steps involved in configuring a collector and attaching data sources. Users start by accessing the collector screen through the UI and adding a new collector, providing necessary configuration details such as collector name, Os type and proxy details. Upon submitting, users are seamlessly redirected to installation steps, facilitating the installation script and other required parameters.

2. Regarding data source configuration, the diagram shows two approaches: attaching data sources before or after collector installation on the host machine. To add a new data source, users visit the data source screen, Add new data source and provide details like data source name, owner name, owner Email Id and Collector (with which data source attached), after clicking on NEXT button user will redirects to Configure Inputs step, in which user provide details like: Log Source, Format, Mode, Port and custom tags. After moving ahead users have the option to filter data based on specific keywords, either through inclusion or exclusion and submit the data source configuration.

4.3 SEQUENCE DIAGRAM

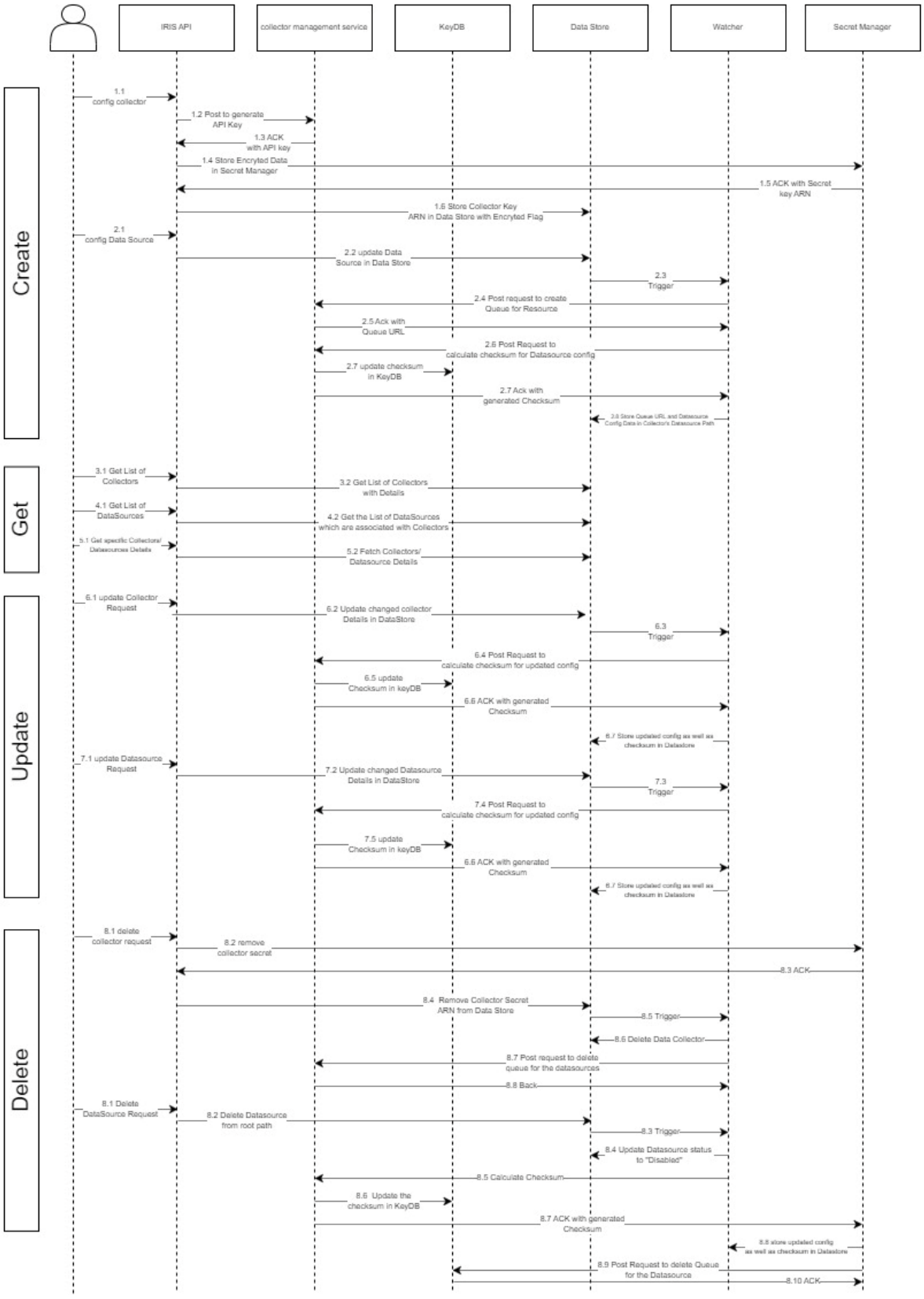


Fig 4.4 Sequence Diagram

Description:

- **Create: Configure Data Collector and Data Source:**
  - This sequence illustrates the steps involved in configuring a new Data Collector and associated Data Source.
  - The User sends a request to the system to create a new Data Collector.
  - The system acknowledges the request and prompts the User to provide configuration details.
  - The User provides configuration details, such as data collection parameters and source information.
  - The system validates the provided information and creates the Data Collector.
  - Next, the User requests to configure a Data Source for the newly created Data Collector.
  - The system acknowledges the request and prompts the User to specify the Data Source details.
  - The User provides the necessary information
  - The system validates the provided details and associates the Data Source with the Data Collector.
- **Get: Retrieve Data Source List and Details:**
  - This sequence demonstrates the process of retrieving a list of Data Sources associated with a Data Collector and fetching their details.
  - The User sends a request to the system to retrieve the list of Data Sources associated with a specific Data Collector.
  - The system acknowledges the request and retrieves the list of associated Data Sources.
  - Subsequently, the User selects a particular Data Source from the list and requests its details.
  - The system acknowledges the request and retrieves and presents the details of the selected Data Source to the User.
- **Update: Update Data Collector and Data Source Request:**
  - This sequence outlines the steps involved in updating the configuration of a Data Collector and modifying the settings of a Data Source.
  - The User sends a request to update the configuration of a specific Data Collector.

- The system acknowledges the request and prompts the User to provide updated configuration details.
- The User provides the updated configuration information.
- The system validates the provided details and updates the configuration of the Data Collector accordingly.
- Similarly, the User requests to update the settings of a particular Data Source associated with the Data Collector.
- The system acknowledges the request and prompts the User to specify the updated settings.
- The User provides the updated settings.
- The system validates the provided information and updates the settings of the Data Source accordingly.

- **Delete: Delete Data Collector and Data Source Request:**
  - This sequence illustrates the process of deleting a Data Collector along with its associated Data Sources.
  - The User initiates a request to delete a specific Data Collector.
  - The system acknowledges the request and confirms the deletion action.
  - Subsequently, the system proceeds to delete the Data Collector and its associated Data Sources.

4.4 USE CASE DIAGRAM

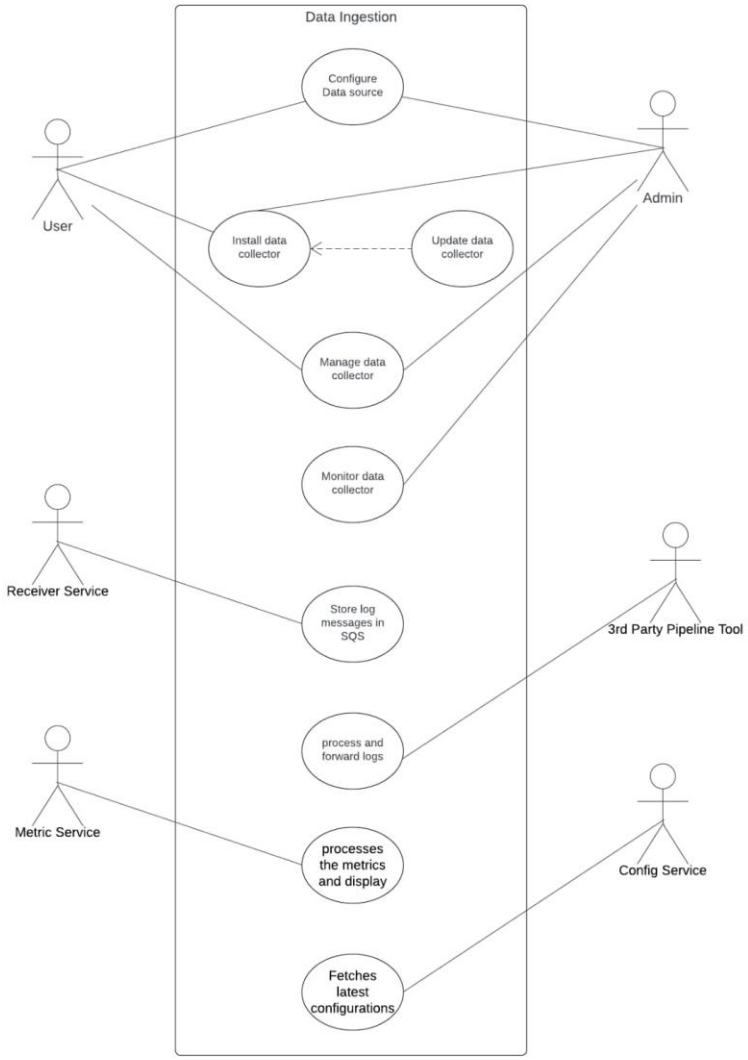


Fig 4.5 Use Case Diagram

Description:

The above Use Case Diagram illustrates the functional interactions within the Data Collection and Integration System. It outlines the system's capabilities and the roles of various actors involved in its operation. The diagram encompasses use cases related to installing and upgrading data collectors, configuring data sources, monitoring system performance, managing data collectors, and interacting with external services.

The "Install Data Collector" use case involves deploying new instances of data collectors to gather logs from diverse sources. This action ensures the continuous flow of data into the system for processing and analysis. Meanwhile, the "Upgrade Data Collector" use case pertains to updating existing data collectors to the latest versions, enhancing system security and functionality.

Configuration of data sources is facilitated by the "Configure Data Source" use case, allowing users to specify parameters and settings for each data source. This ensures that the data collectors collect relevant information in accordance with user requirements. Monitoring system performance is achieved through the "Monitor Data Collector" use case, enabling users to track the status and health of data collectors in real-time.

Administrative tasks are addressed by the "Manage Data Collector" use case, empowering system administrators to start, stop, or restart data collectors as needed. This use case also encompasses other administrative functions necessary for maintaining system integrity and performance. Additionally, the system interfaces with external services such as the Receiver Service (utilizing Amazon SQS) and Third-Party Pipeline Tool, which handle storage and downstream processing of log messages, respectively.

The "Config Service" use case involves fetching and applying the latest configurations, particularly when users modify data source settings. This ensures that the system remains up-to-date and adaptable to changing requirements. Lastly, the "Metric Service" use case focuses on processing data and generating insights, providing users with valuable information for decision-making and performance optimization.

CHAPTER 5: IMPLEMENTATION

5.1 IMPLEMENTATION ENVIRONMENT

The implementation environment for Data Ingestion typically includes the following components:

- **Go SDK**: Data Ingestion is implemented using the Go SDK, which provides a framework for building native applications for web and desktop platforms using a single codebase.
- **Packages**: Data Ingestion mainly utilizes Viper, Mock and Testing package which are used to handle all types of configuration needs and formats, to mock objects and verify calls whether they are happening as expected and to automated testing of Go packages respectively.
- **IDE**: Any IDE that supports Go development, such as Visual Studio Code, GoLand can be used to implement Data Ingestion.
- **Operating system**: Data Ingestion is implemented on any operating system including Windows and Linux.
- **Virtual Machine**: To test proper functioning during development of Data Ingestion, Virtual machines of different OS were used.
- **Testing framework**: To ensure the quality of Data Ingestion, automated testing is performed using testing frameworks such as Go's third-party libraries like httpmock or testify.

5.2 SERVICES SPECIFICATIONS

Services in Data Ingestion are mainly divided in 2 parts,

- Data Collector service
- Data Receiver service

5.2.1 Data Collector Service

- **3rd Party Pipeline Tool**:

  An agent is used for log processing and forwarding the collected data. It parses the data, filters it and forwards the logs. The agent can collect the logs of the type syslog currently. The agent supports log parsing and filtering capabilities, allowing us to extract structured data from unstructured log messages and filter logs based on specific criteria. Agent is lightweight and efficient, with minimal resource consumption and also the agent offers compatibility with various distributions of linux and extends the support to windows.

- **Metrics Scheduler**:

  The metric scheduler is used to check the health status of the agent over a regular period of time. It pulls the metrics from the data forwarder and sends it through the agent to the metrics service.

- **Config Scheduler:**

  The scheduler will generate a checksum based on the configuration and cross-reference it with the config service for validation. If different, it fetches and replaces the configuration, orchestrating the third-party tool's restart. Upon reload confirmation, the service updates the status. If there is no difference, it notifies the configuration's up-to-date status, ensuring smooth configuration synchronization with minimal disruption. This ensures that the scheduler operates with the most up-to-date configuration, enhancing reliability and consistency in its tasks.

5.2.2 Data Receiver Service

- **Metric Service**

  In this architecture, the client metric service sends resource utilization data to the receiver metric service through an API gateway. Before transmission, the receiver service validates API calls for security. Upon reception, it processes the metrics and displays them using a client-specified monitoring tool. This ensures secure data transfer and enables clients to make informed decisions with real-time performance insights.

- **Receiver Service**

  The receiver service is crucial in the system, collecting logs from a data forwarder via an API gateway. It verifies API calls before enriching the logs with metadata for clarity. The enriched logs are then stored in AWS' SQS for efficient handling and storage. This process ensures secure access, enhanced comprehension, and seamless retrieval for analysis, optimizing system performance.

- **Config Service**

  The configuration service manages on-premises device configurations via its portal. The service compares the on-premises configuration checksum with the latest. If different, it fetches the latest configuration and sends it to the config scheduler.

- **Agent Management Service**

  The agent management service orchestrates tasks like API key generation, checksum calculations, and SQS queue management. It communicates with the config API to update configurations and trigger queue creation for new data sources. Existing sources prompt updates, ensuring accurate checksums. This process ensures smooth configuration integration and reliable data integrity.

- **Config APIS**

  The config API is central, enabling users to update or create data sources and forwarders. It guides users through configuring and deploying forwarders. With its user-friendly interface and robust functionality, it enhances data management efficiency, fostering system agility and adaptability.

- **Watcher**

  The watcher component responds to changes in the key-value store configuration, ensuring real-time synchronization and triggering checksum calculations for data integrity. By monitoring and facilitating actions based on configuration updates, it enhances system reliability and responsiveness, ensuring accurate reflection of the latest settings across all components.

5.2.3 Cloud Managed Services

- **Secret Manager**

  The secret manager serves as a secure repository for storing all configuration API keys, ensuring the protection of sensitive information. By leveraging the secret manager, organizations can safeguard their API keys from unauthorized access and potential security breaches. This centralized storage solution offers robust encryption and access control mechanisms, mitigating the risk of key exposure and unauthorized usage.

- **Lambda Authorizer**

  A Lambda authorizer is useful if you want to implement a custom authorization scheme. When a client makes a request to one of your API's methods, API Gateway calls your Lambda authorizer. A Lambda function validates the API key of a data forwarder with the receiver. If the request authenticates the process continues otherwise it will throw an error to the data forwarder.

5.3 OUTCOMES

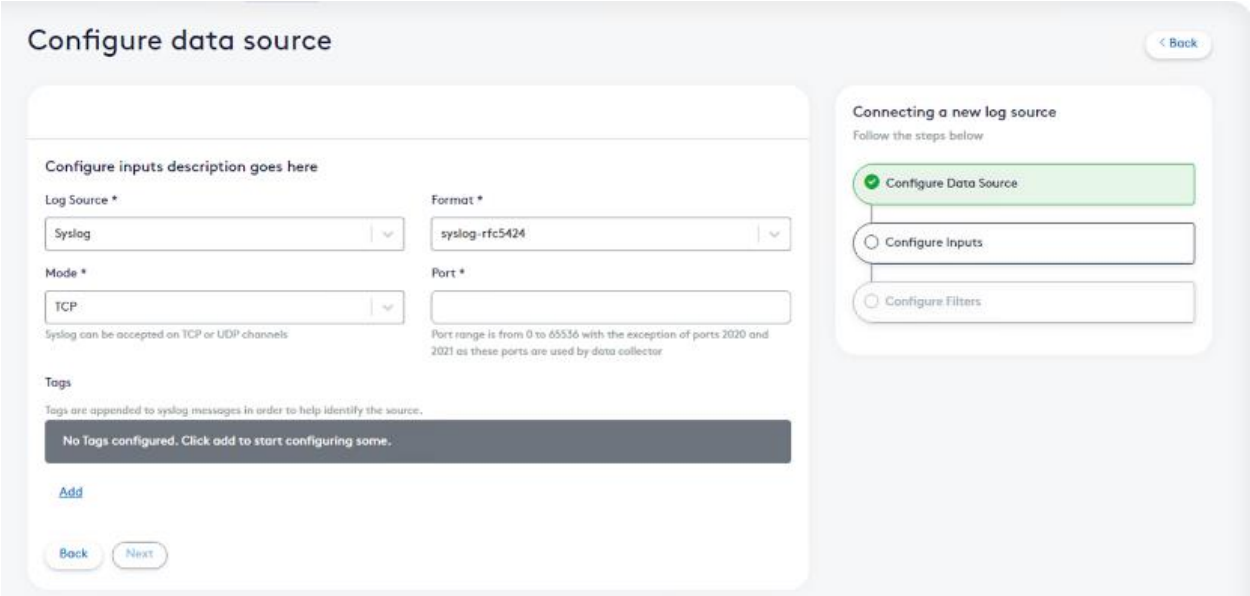


Fig 5.1 Configure Datasource

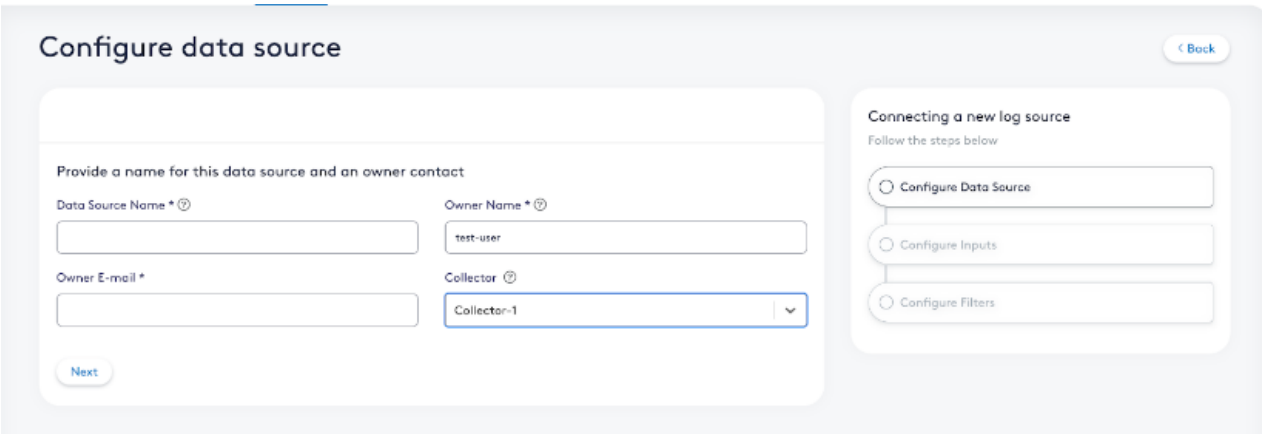


Fig 5.2 Configure Datasource

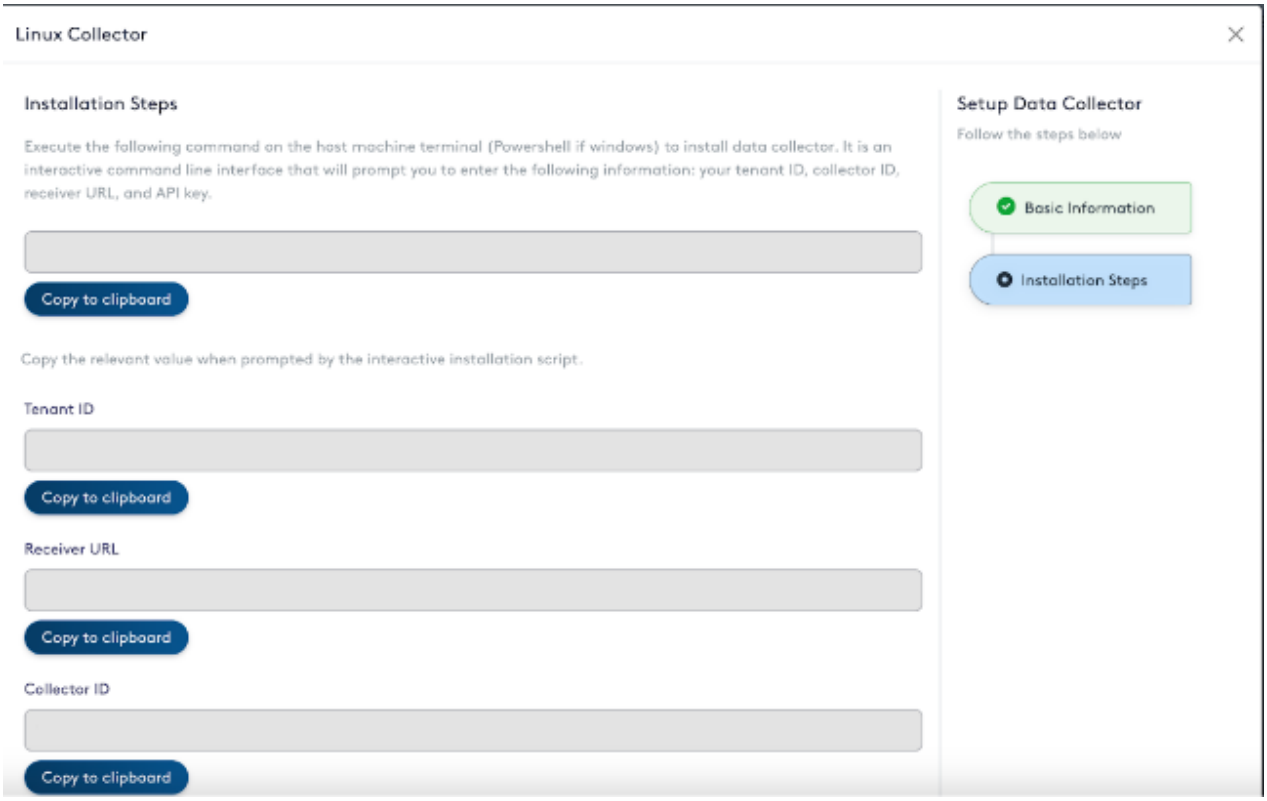


Fig 5.3 Setup Data Collector

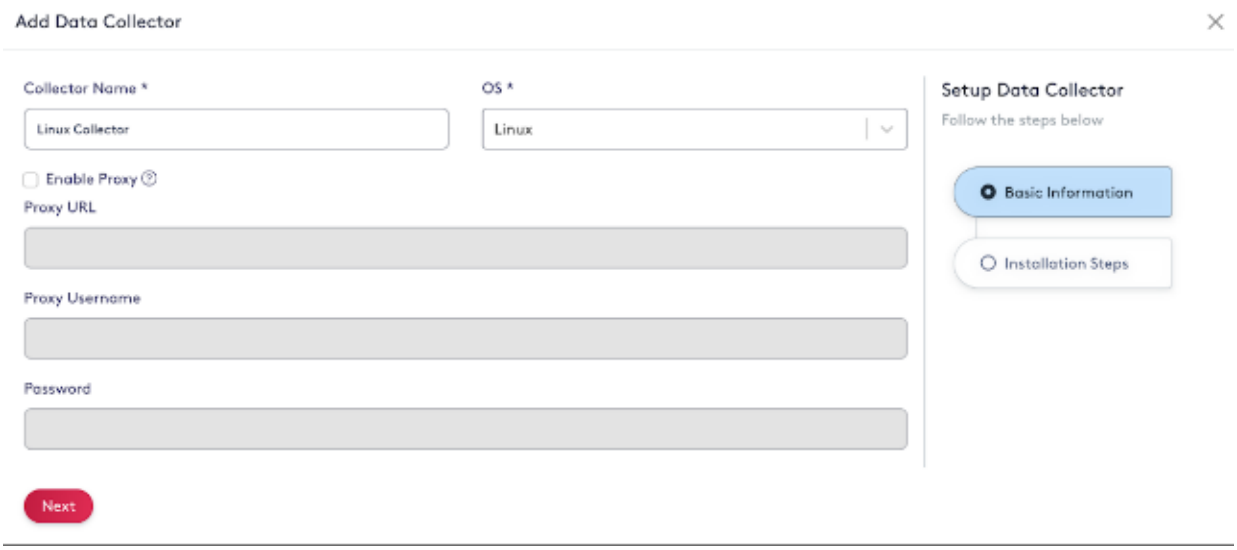


Fig 5.4 Configure Data Collector

CHAPTER 6: TESTING

6.1 TESTING PLAN

- **Functional Testing:** This type of testing ensures that the Collector functions correctly and meets the requirements of the user. It involves testing individual features such as collecting the data, forwarding the data, and sending metrics to third party tool.

- **Usability Testing:** This type of testing focuses on the user experience and interface of the data collector. It involves testing the data collector ease of use and overall user experience.

- **Compatibility Testing:** This type of testing ensures that the data collector is compatible with different systems or environment (Linux and windows).

- **Performance Testing:** This type of testing focuses on the performance of the Data Collector, including its rate of data ingestion. It involves testing the collector under various load conditions to ensure that it can handle a high volume of data.

- **Regression Testing:** This type of testing ensures that the data collector continues to function correctly after updates or changes are made. It involves testing the collector's existing features to ensure that they have not been impacted by new changes.

6.2 TEST RESULTS AND ANALYSIS

Table 6.1 Test Results and Analysis

Test case ID	Component	Summary	Expected Output	Actual Output	Status
T1	Functional	Verify user is able to login to the existing configured tenant.	User is navigated to the next page.	User is navigated to the next page	Pass
T2	Functional	Verify user is able to navigate throughout the data collectors page .	User should get the list of data collectors available.	User should get the list of data collectors available.	Pass
T3	Functional	Verify that user can click on the collector and edit the configurations.	User is redirected to form for editing configuration of collector.	User is redirected to form for editing configuration of collector.	Pass
T4	Functional	Verify that updated configurations are identified and fetched by the adapter service.	The configuration changes should be reflected to the configuration Scheduler.	The configuration changes are reflected to the configuration Scheduler.	Pass
T5	Functional	Verify that the installation script is updated as per the selected OS.	Different scripts are produced for different OS	Different scripts are produced for different OS	Pass
T6	UI	Data source overview	User should get an overview of data source configuration details.	User gets a overview of data source configuration details.	Pass
T7	UI	Button for data source addition.	User should be able to add a new data source from the available data source list.	User should be able to add a new data source from the available data source list.	Pass
T8	functional	Verify user is redirected to next page to configure the new data source.	Once user selects data collector, the data source configuration page opens.	Once user selects data collector, the data source configuration page opens.	Pass
T9	UI	Verify required fields	User cannot progress further without filling the fields.	User cannot progress further without filling the fields.	Pass

T10	UI	Verify that all the necessary scripts are provided to the user for collector setup.	The user is provided the script to be run in the terminal and the credentials for set up.	The user is provided the script to be run in the terminal and the credentials for set up.	Pass
T11	UI	Selection of log types.	User is provided options to select the log types.	User is provided options to select the log types.	Pass
T12	Functional	Verify that adapter is not loaded in case of no updates.	The 3rd party collector tool is not reloaded if the configurations are not updated.	The 3rd party collector tool is not reloaded if the configurations are not updated.	Pass
T13	Functional	Verify log parsing	The received logs are successfully parsed to the ingestion service with necessary information.	The received logs are successfully parsed to the ingestion service with necessary information.	Pass
T14	Functional	Verify the monitoring service data.	The system monitor should be able to send relevant metrics to the monitoring service.	The system monitor is able to send relevant metrics to the monitoring service.	Pass
T15	UI	Verify data source updation on UI.	The updated log type should be displayed on the UI.	The updated log type is displayed on the UI.	Pass
T16	Functional	Disable collector	The user should be able to disable the collector and all the connected data sources should be disabled.	The user should be able to disable the collector and all the connected data sources should be disabled.	Pass
T17	UI	Data source deletion	When data source is deleted its configuration schema should be deleted.	When data source is deleted its configuration schema should be deleted.	Pass

CHAPTER 7: CONCLUSION AND DISCUSSION

7.1 OVERALL ANALYSIS OF INTERNSHIP

Based on the objectives and scope of the Software developer internship at Crest Data Systems, it appears to be a valuable opportunity for individuals looking to develop their skills in Software development. The internship offers hands-on experience in building cross-platform web applications, working on real-world projects, and receiving mentorship from experienced developers.

Additionally, the internship provides access to the latest tools and technologies in the industry and an opportunity to develop critical skills such as communication, problem-solving, and critical thinking in a professional setting.

Overall, the Software developer internship at Crest Data Systems seems to be a well-structured program that could provide a solid foundation for a career in Software development, and potentially even lead to future employment opportunities with the company.

7.2 PHOTOGRAPHS AND DATE OF VISIT BY INSTITUTE MENTOR

Date of Visit : 15 April 2024

Fig 7.1 Visit By Institute Mentor

7.3 DATES OF CONTINUOUS EVALUATION (CE-I , CE-II AND CE-III)

Continuous Evaluation-I was done on 03 February 2024 by internal guide Dr. Hiteishi Diwanji, I presented my Project ideas in offline mode.

Continuous Evaluation-II was done on 02 March 2024 by internal guide Dr. Hiteishi Diwanji, I presented diagrams and some other questions asked by ma'am in offline mode.

Continuous Evaluation-III was done on 06 April 2024 by internal guide Dr. Hiteishi Diwanji, I presented modified diagrams and SRS of the project in offline mode.

7.4 PROBLEM ENCOUNTERED AND POSSIBLE SOLUTION

- **Complexity:** A lot of different components and services are involved while designing & developing an architecture which is to be used by lot of users. With component planning & management, I established modularity in component design. This also helped in error isolation and brought reusability to the project.
- **Collector Down :** While ingesting amount of data which is more than the capacity of receiver, collector might get down. So, sometimes this can be case and by seeing it's report we can solve problem.
- **Resource Utilization :** When amount of data to be ingested become too large, then 3rd party pipeline tool heavily consumes cpu and memory usage.

7.5 SUMMARY OF INTERNSHIP WORK

During my internship, I acquired a wealth of technical expertise in Golang concepts. Additionally, I honed non-technical skills such as teamwork and effective communication. The experience also afforded me the opportunity to develop soft skills like professional behaviour, etiquette, and work ethics, which are invaluable in any professional setting. This internship served as a pivotal step in my transition from being an engineering student to becoming a software engineer. Working on a project provided invaluable insights into client management, meeting deadlines, and enhancing existing code bases.

7.6 LIMITATION AND FUTURE ENHANCEMENT

7.6.1 Limitations:

- Collector agent is installed on premise. Thus, leading to the threat of single point of failure.
- Only syslog formats are supported for the logs.
- Raw logs are sent to the receiver service.
- Product UI is complex.

7.6.2 Future Enhancement:

- Expand the collector agents to avoid the single point of failure.
- Provide some user-friendly way to install the collector.
- Extend the functionality to provide support for following log format:
    - JSON
    - Windows Event
    - TCP

Extend the functionality to send some metadata along with the raw logs.

Facilitate intuitive user interface (UI) design to ensure clients can easily comprehend the system.

References

- **Go Documentation:** https://go.dev/learn/
- **Go Programming Language:** https://tip.golang.org/doc/go1.20
- **AWS Documentation:** https://aws.amazon.com/documentation/
- **Syslog Documentation:** https://www.solarwinds.com/syslog/
- **SQS Documentation:** https://docs.aws.amazon.com/sqs.html
- **YAML Documentation:** https://yaml.org/
- **Github Action Documentation:** https://docs.github.com/en/actions