

Requirements Doc

Infogrep must be (core goals)

- When service is deployed and configured, service can be operated without any network traffic leaving the company network
- Infogrep services must be scalable
- Configurable
- Files, vectors, and chatrooms must be stored locally

Infogrep must be (constraints)

- Interfacing with the frontend does not cause the frontend to be hung on blocking operations
- Backend services must follow micro services architecture.
- Backend services other than the AI service must provide responses within 50 ms
- File service must be designed so that it is easy to add support for new storage options
- Frontend must be able to run on all major web platforms
- Infogrep must support OAuth login and local username and passwords
- Infogrep must support one click deployment
- Infogrep be able to handle PDF and HTML documents and Confluence Integrations. Other file types are a nice to have
- Document parsing must be done on device
- Infogrep must support multiple LLMs and embeddings from multiple providers
- Infogrep must support admin control over the service to configure LLMs, embeddings, stored files, and chatrooms
- Infogrep must support chatrooms where documents can be uploaded. Chatrooms must be visible only to the user who made it and admins
- Users must be able to view uploaded files, download them, and delete them.
- Uploaded documents must get parsed by the AI service

Design Document

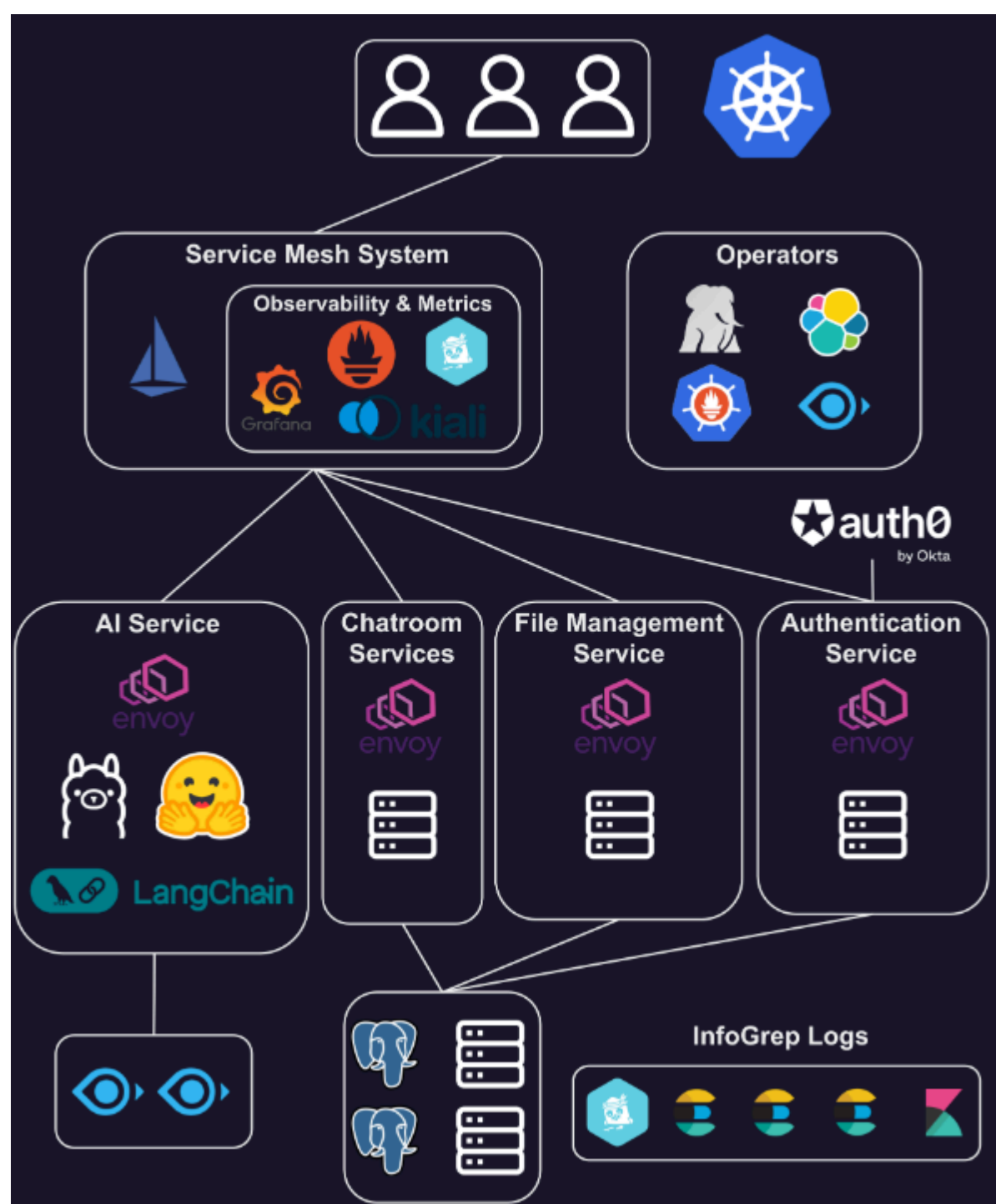
This document describes InfoGrep's raison d'être, its architecture for frontend and backend systems and key milestones and goals envisioned by the team.

Preamble

With the exploding popularity of LLM models and chat-like platforms, the enterprise sector have began to take notice and leverage this powerful tool for internal productivity and innovations. Oftentimes, this is done through in-house development of a chat bot interfacing with work platforms like Slack or Microsoft Teams, with the bot pulling knowledge from specific platforms like Confluence or documents. Due to its in-house nature, work is duplicated across companies and teams who each build their own solutions and re-invent the wheel. InfoGrep aims to be an extensible and open framework that covers common use-cases and allows companies to instead build and innovate on top of us.

System Architecture

InfoGrep is comprised of two main parts, the frontend and backend who each contain additional components.



Backend

The backend leverages Kubernetes, an container orchestration system that deploys, updates and scales containerized applications that is programmable through YAML configurations. Containerized applications work well with scaling and fault tolerance since Kubernetes allows us to start multiple "pods" of the same container service. With this in mind, InfoGrep backend is split into 4 services that each handle a unique subset of features: AI, Chatroom, File Management and Authentication.

Each service is written in Python and Uvicorn, an asynchronous framework for web servers. From the point of view of the service, it talks to a 'centralized' PostgreSQL server that is in fact scaled by Kubernetes through the Cloud Native Postgres operator.

Some service-specific components can be found in AI and Authentication. The AI service leverages Langchain, a library that abstracts common LLM operations like chat history and using an LLM models to generate a user response. With Langchain, InfoGrep is able to support both offline and online AI generation. Offline support means the LLM model is run on the enterprise servers and is supported through Langchain's Ollama APIs. InfoGrep's online generation leverages Langchain API abstractions for popular online LLM API providers like OpenAI, Cloudflare, provided that the user specifies an API key through the admin frontend.

The authentication service supports the OAuth 2.0 protocol, thus allowing for single sign-ons from enterprise auth providers like Okta. This is especially convenient for enterprise adoption since nearly all enterprise workplaces have some form of single sign-on already implemented to control access to internal resources.

Frontend

The frontend responsibility is to provide a working demonstration of the backend's capabilities. This includes non-privileged user features like creating a chatroom, chatting with the LLM models, uploading files, etc, while admin features include setting API keys for online LLM APIs, managing and deleting chatrooms system-wide and creating/deleting users.

Internally, the frontend is built with React so each feature can be split into smaller modular components. Redux is also used for a site-wide global state store which simplifies state management when there are many components who need to keep track of state that is often shared. The UI is built with components from Material UI, an open source UI library that includes common components like buttons, text inputs and typography for React.

On a code level, the non-privileged parts and the admin control panel are almost entirely isolated in terms of Redux states. This is because the set of features these pages have are disjoint.

Milestones and Goals

InfoGrep aims to solve common use cases so companies can build on top of us. As such, there are certain milestones and goals that will let us achieve that vision:

- Support completely offline AI generation for data sovereignty and security
 - This is fully supported through Langchain's integrations for Ollama, a service that is deployable fully on-premise and which allows the admin to download models that can be run entirely offline.
- Support extensibility on a framework level so companies can build on top
 - This is supported in different ways; Infogrep has been open sourced on Github, which allows anyone to deploy it for themselves and make modifications. Moreover, InfoGrep supports event-based webhooks so companies can hook into important InfoGrep events like user sending a message and react appropriate by writing their own Lambda functions.
- Support multi-media inputs from users for a rich experience
 - InfoGrep supports users sending plain text messages as well as uploading files to enhance the chatbot's knowledge and to reduce the chance of AI hallucination.
- Support suitable enterprise-specific features for easy adoption
 - InfoGrep supports enterprise single sign-on via the OAuth 2.0 protocol for easy employee access. It also contains an admin control panel with sign-in auditing history to enable a sufficient level of control that enterprise workplaces require.

InfoGrep User Manual APRIL2025

Team 28

- Qirong He, q48he
- Tanbir Banipal, tbanipal
- Gary Li, gary.li1
- Jacky Xu, j557xu
- Thomas Nie, wtnie
- Jacob Scott, j42scott

April 14, 2025

Table of Contents

[Table of Contents](#)

[List of Figures](#)

[Chapter 1 - Introduction](#)

[1.1 Product Overview](#)

[1.2 Setup Overview](#)

[1.2.1 Directory Structure](#)

[1.2.2 Prerequisites](#)

[1.2.3 Download the InfoGrep Deployment Code](#)

[1.2.4 Configuration](#)

[1.2.5 Deployment](#)

[1.2.6 Accessing the InfoGrep Application](#)

[1.2.7 Post Deployment Check](#)

[1.3 First Run Example](#)

[Chapter 2 - Conventions](#)

[2.1 Notational Conventions](#)

[2.2 Terms](#)

[2.3 Abbreviations](#)

[2.4 Basic User Interface Goals](#)

[2.5 First Sample Redone](#)

[2.6 Organization of this Manual](#)

[Chapter 3 - Domain Model, Use Case Diagram, System Architecture Diagram, and Assumptions, Exceptions, and Variances](#)

[3.1 Domain Model](#)

[3.2 Use Case Diagram](#)

[3.3 Assumptions, Exceptions, and Variances](#)

[3.3.1 SA \(*System Administrator*\) Assumption](#)

[3.3.2 Application User Assumption](#)

[3.4 Exceptions](#)

[3.4.1 SA \(*System Administrator*\) Exceptions](#)

[3.4.2 Application User Exceptions](#)

[3.5 Variations](#)

[3.6 Exceptional Scenarios and Alternative Scenarios Prevention and Handling for System Administrator](#)

[3.6.1 Human-Level Exceptional Scenarios Prevention and Handling](#)

[3.6.2 Infrastructure-Level Exceptional Scenarios Prevention and Handling](#)

[3.6.3 Human-Level Alternative Scenarios Prevention and Handling](#)

[3.7 Exceptional Scenarios and Alternative Scenarios Prevention and Handling for Application User](#)

[3.7.1 Human-Level Exceptional Scenarios Prevention and Handling](#)

[3.7.2 System-Level Exceptional Scenarios Prevention and Handling](#)

[3.7.3 Human-Level Alternative Scenarios Prevention and Handling](#)

[3.7.4 System-Level Alternative Scenarios Prevention and Handling](#)

[3.8 System Architecture Diagram](#)

[Chapter 4 - Basic Use Cases](#)

[4.1 InfoGrep Configuration](#)

[4.1.1 InfoGrep Helm Charts and Default Value](#)

[4.1.2 Updating Infogrep Configurations](#)

- [4.2 InfoGrep Deployment](#)
 - [4.2.1 InfoGrep Deployment Command](#)
 - [4.2.2 InfoGrep Deployment Verification](#)
 - [4.2.3 Uninstalling Infogrep](#)
- [4.3 User Account](#)
 - [4.3.1 The User Login Page](#)
 - [4.3.2 The User Settings Page](#)
- [4.4 Managing Chatrooms](#)
 - [4.4.1 The Chatroom Manager](#)
 - [4.4.2 Creating a Chatroom](#)
 - [4.4.3 Deleting a Chatroom](#)
- [4.5 Chatroom Interface](#)
 - [4.5.1 Sending a message](#)
 - [4.5.2 Viewing the Chat History](#)
- [4.6 Document Collection](#)
 - [4.6.1 Adding a Document to the Document Collection and Adding a Integration](#)
 - [4.6.2 Viewing Uploaded Documents](#)
 - [4.6.3 Removing Uploaded Documents](#)
- [4.7 Admin Controls](#)
 - [4.7.1 Managing Users](#)
 - [4.7.2 Provider Settings](#)
 - [4.7.3 Managing LLMs](#)
 - [4.7.4 Managing Files](#)
 - [4.7.5 Creating Webhooks](#)

[Chapter 5 - Application User Trouble Shooting and Tips](#)

- [5.1 Troubleshooting](#)
- [5.2 Tips](#)

[Chapter 6 - Limitations and Bibliography](#)

- [6.1 Limitations](#)
- [6.2 Bibliography](#)

List of Figures

- 1.1 Login screen layout of InfoGrep
- 1.2 Screen layout of the main screen for InfoGrep
- 1.2.1.1 Infogrep Root Directory Layout
- 1.2.1.2 Infogrep Charts Directory Layout
- 1.2.1.3 Infogrep Dashboard Directory
- 1.2.1.4 Infogrep Hack Directory
- 1.2.1.5 Infogrep Templates Directory
- 3.1 Domain Model Diagram
- 3.2 Use Case Diagram
- 3.8 System Architecture Diagram
- 4.0.1. Infogrep Docs Example
- 4.1 The User Login Page
- 4.2 The User Settings Page
- 4.3 The Chatroom Manager
- 4.4 Creating a new Chatroom from the Chatroom Manager
- 4.5 Deleting a Chatroom from the Chatroom Manager
- 4.6 Sending a Message in a Chatroom
- 4.7 The Chatroom History
- 4.8 Sending Suggested Input

- 4.9 Rating an InfoGrep Reply
- 4.10 Regenerating an InfoGrep Reply
- 4.11 Highlighted Message Item
- 4.12 Highlighted Item View
- 4.13 Uploading a Document
- 4.14 Document Collection Widget
- 4.15 Removing Documents from a Document Collection
- 5.1. Infogrep Logs
- 5.2 Infogrep Kiali
- 5.3 Infogrep Jaeger

Chapter 1 - Introduction

1.1 Product Overview

InfoGrep is an open-source *RAG (Retrieval Augmented Generation)* framework which provides an enterprise solution for integrating domain-specific *AI (artificial intelligence)* seamlessly into workplace environments.

InfoGrep empowers companies to configure customized *large language models (LLMs)* into the platform for employee use, and helps users with questions about files or links that they input. As the name suggests, it is akin to the grep command for the info the user submits to each chatroom. The solution natively supports all standard file types and offers integration for tools like Confluence, JIRA, and various storyboarding platforms.

InfoGrep is provided as a versatile framework, it will be available as a standalone service and a scalable, production-ready distributed deployment. InfoGrep runs on MacOS and Linux environments, utilizing Electron for the front end, and mainly Python for the back end.

1.2 Setup Overview

1.2.1 Directory Structure

Root Directory








 .github
 infogrep-charts
 scripts
 .gitattributes
 .gitignore
 README.md
 config.template.yaml

Figure 1.2.1.1 Root Directory Layout

Infogrep Charts Directory







Name
 ..
 dashboards
 hack
 templates
 Chart.yaml
 values.yaml

Figure 1.2.1.2 Infoigrep Charts Directory Layout

In the root directory

- `.config.template.yaml` is a provided sample setup configuration file that users could override with their own customized values when deploying InfoGrep.
- `README.md` in the root directory contains the setup & testing instructions.

- `.gitignore` and `.gitattributes` are GitHub configuration files for source control management.
- `/.github` contains the Helm charts release publish continuous integration.
- `/scripts` contains scripts for deploying, updating, and uninstalling Infogrep.
- `/infogrep-charts` contains Helm charts and configurations of an Infogrep Deployment.

In the `infogrep-charts` directory

- `Charts.yaml` is A YAML file containing information about the Infogrep charts
- `values.yaml` is a YAML file containing configurations that an Infogrep deployment will use.
- `/dashboards` contains Infogrep’s Grafana observability dashboards.








Name	
	..
	extension.json
	mesh.json
	performance.json
	pilot.json
	service.json
	workload.json

Figure 1.2.1.3 Infogrep Dashboard Directory

- `/hack` contains helper configuration file used in the scripts




Name	
	..
	mesh-config-default.yaml
	mesh-config-ingressgateway.yaml

Figure 1.2.1.4 Infogrep Hack Directory

- `/templates` contains the actual templated Kubernetes Helm Charts of Infogrep.













Name
 ..
 infogrep-configs
 infogrep-ingress
 infogrep-service
 infogrep-sts
 infogrep-telemetry
 .helmignore
 NOTES.txt
 _common.tpl
 _helpers.tpl
 _versions.tpl
 namespace.yaml

Figure 1.2.1.5 Infogrep Templates Directory

1.2.2 Prerequisites

- **System Requirements:**
 - **Operating System:** MacOS or Linux
 - **Memory:** Minimum 8 GB RAM recommended
 - **Storage:** At least 20 GB free space for installation, additional space for data storage
- **Software Requirements:**
 - **Kubernetes:** A running Kubernetes cluster is available and configured.
 - **Helm:** Version 3.0 or later.
 - **Istioctl:** The CLI tool for managing and controlling the Istio service mesh.
 - **Longhorn (or any storage controller):** For managing persistent volume for stateful sets.

|
 Note: Ensure that every dependencies is installed before proceeding with the setup.

1.2.3 Download the InfoGrep Deployment Code

1. Go to the official InfoGrep repository or website at <https://github.com/SE-Exort/InfoGrep-Deployment>.
2. If Git is installed, run `git clone https://github.com/SE-Exort/InfoGrep-Deployment.git` to download the repository, otherwise, download the zip at <https://github.com/SE-Exort/InfoGrep-Deployment/archive/refs/heads/main.zip>.

1.2.4 Configuration

- **Environment Configuration:**

- Navigate to the Infogrep root directory and locate the `config.template.yaml` file.
- Modify the values as needed to fit your environment (e.g., ingress type, deployment type, LLMs, resource needed, etc.).

1.2.5 Deployment

- **Helm Deployment:**

- Under the root directory, install Infogrep with

```
bash scripts/install.sh
```

1.2.6 Accessing the InfoGrep Application

- Wait for Infogrep to finish installation.
- You're now ready to start using InfoGrep. Visit the InfoGrep UI application URL from the `infogrep-frontend` load balancer address. You can obtain this address with `kubectl get svc -n infogrep`

1.2.7 Post Deployment Check

- Ensure Kubernetes InfoGrep resources are running correctly:

```
kubectl get all -n infogrep
```

- Verify that InfoGrep is responding correctly by logging in as the administrator.

1.3 First Run Example

To start, an application user should follow the setup instructions and deploy InfoGrep on a server with appropriate access. The user can now visit the InfoGrep application's URL and land on the *login page*. Then the user must log into the app to access their document collections and chat rooms. They have the option to sign up on the login screen layout or log into their account. To sign up, the user must:

1. Left-click on the **Email textbox** then write in a valid email address, following the regex

```
^[a-zA-Z0-9-_%+]+@[a-zA-Z0-9-]+\.[a-zA-Z]{2,}$ .
```

2. Left-click on the **Password textbox** then write a password that exceeds 8 characters.
3. Left-click on the **Signup button**.

Both the **Login** and **Signup buttons** will lead them to the *main layout* of the program. The screen layout of the login screen is found in Figure 1.1.

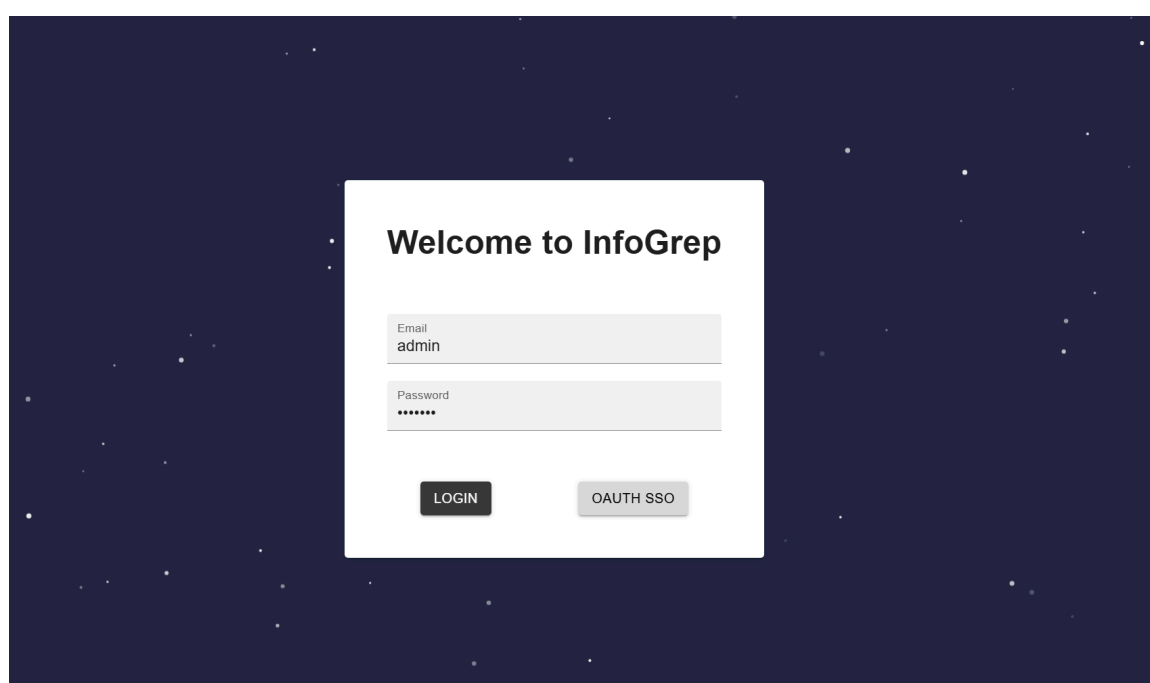


Figure 1.1 Screen layout of the login screen for InfoGrep

Next, a user can create a **Chatroom**, by using the **New Chat button** in the top left of the *main screen layout* which is found in Figure 1.2. Then a user can ask a question to InfoGrep using the **Chatroom textbox** at the bottom of the screen.

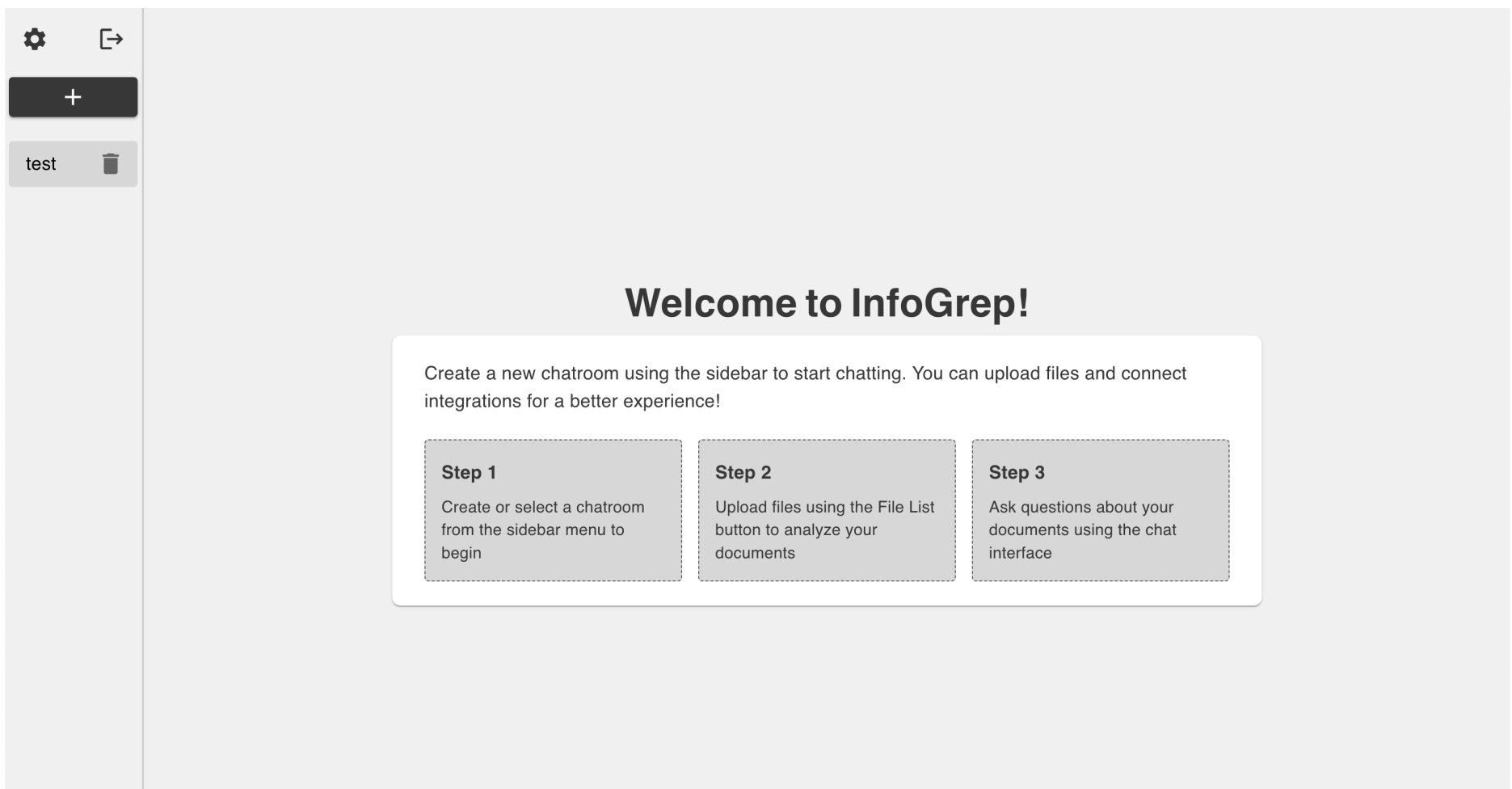


Figure 1.2 Screen layout of the main screen for InfoGrep

Chapter 2 - Conventions

2.1 Notational Conventions

The following text conventions are used in the manual:

- Sans-serif is used for normal text.
- *Sans-Serif Italics* is used for emphasis and new terms in normal text.
- **Sans-serif bold** is used for UI elements and sub-section titles.
- MenuA→MenuItemB is used to specify the menu item MenuItemB in the menu MenuA.
- `Sans-Serif Code Block` is used for any command snippets and file paths.

2.2 Terms

The following terms are used throughout the manual:

- InfoGrep – the name of the program.
 - Also, the term is used to refer to the *AI* system.
- InfoGrep Application User - the user that uses InfoGrep's main application
- System Administrator - abbreviated as SA, the person responsible for configuring and managing a company's entire infrastructure, including all of the hardware, software, and operating systems that are necessary to support the running of the business.
- InfoGrep SA - the user that configures, sets up, and deploys InfoGrep into their organization's infrastructure.
- InfoGrep servers – backend servers that handle all network transactions between the application user and the InfoGrep system.
- AI - AI or Artificial Intelligence is what makes InfoGrep possible by using a large language model to answer the user's questions.
- screen layout – a collection of widgets on the screen, shown in Figure 1.1, and Figure 1.2
- Login screen layout – made up of the **Login widget** and the InfoGrep logo.

- Main screen layout – made up of the Chatroom and the Chatroom Manager
- Login widget – made up of the **Email textbox**, **Password Textbox**, the **Single Sign On button**, and the **Login button**.
- Email textbox – a textbox in which the user enters their email to sign up or log in
- Password textbox – a textbox in which the user enters their password to sign up or log in
- Single Sign On button – a button that signs the user up for an account with the text within the **Email textbox** for their email, and the text within the **Password textbox** for their password.
- Login button – a button that logs the user into the account with the text within the **Email textbox** for their email, and the text within the **Password textbox** for their password.
- Chatroom – a metaphorical room in which the user is with the InfoGrep servers that contain the AI. It contains a **Message History**, **Document Collection Widget**, **Regenerate Response Button**, **Suggestion Button**, and **Chatroom textbox**.
- Message History – A scrolling display of the past messages from the user, and the *InfoGrep reply(s)*.
- Document Collection Widget – a container that displays the documents used for the current *document collection* and other document collection available
- Document Collection - a list of documents the user uploaded and chose to use for the current generation.
- Regenerate Response Button: a button that regenerates the last InfoGrep reply(s).
- Suggestion Button: a button with a pre-filled input text, the user can click and send this input directly to InfoGrep.
- Chatroom textbox – a textbox that allows the user to send messages to the AI.
- Chatroom manager – used to navigate between previously created chatrooms, create new chatrooms, and delete previously created chatrooms.
- Chatroom manager toggle – used to toggle the **Chatroom Manager** off-screen, and back on screen if the **Chatroom Manager** is currently off-screen.
- left mouse click – abbreviated as *LMC*, a left mouse click.
- right mouse click – abbreviated as *RMC*, a right mouse click.
- double click – two *LMCs* not more than 1 second apart.
- pop-up menu – a menu that is opened on top of the *main screen layout*.
- InfoGrep reply – the reply InfoGrep's servers provide to the InfoGrep application which is viewable in the **Message history**.

2.3 Abbreviations

- AI – Artificial Intelligence
- RAG - Retrieval Augmented Generation
- SA - System Administrator
- LLM – Large Language Model
- LMC – Left mouse click
- RMC – Right mouse click
- UI – User interface
- OS – Operating System
- ELB - AWS Elastic Load Balancing
- EKS - AWS Elastic Kubernetes Service

2.4 Basic User Interface Goals

The goals for the user interface are to allow users to interact with the AI assistance we provide seamlessly, allow users to manage the previous **Chatrooms** they've created, and manage the **Document Collections** they are currently using.

- The **Chatroom manager** allows users to save their Chatrooms from previous sessions with the application so that they can reread the **Chatroom history**.
- The **Document Collection Widget** allows users to view the current document used for generation, and switch to a previously saved collection, or create a new collection of documents.
- The **Chatroom** allows users to interact with their AI chatbot including:
 - Sending text inputs to the chatbot.
 - Scroll through the Message History
 - Click and send a suggested text input to the chatbot.
 - Click and regenerate the last InfoGrep reply(s).
- The login screen allows users to save their chatrooms from previous sessions in between devices, if the user logs in with the same account on a second device, they will have the same state of account as the first device.

2.5 First Sample Redone

As mentioned in Section 1.2, there are always alternative ways to access the AI chatbots. Besides signing up as described in Section 1.2, you can also utilize the

Login button and the **Chatroom manager**

to continue a previous conversation with the chatbots.

To start, a user should launch the InfoGrep program on their device. Then, the user must log into the app to access their chatrooms. On the login screen, they have the option to sign up or log into their account.

To log in, the user must:

1. Have already signed up with an email and password combination.
2. Left-click on the **Email textbox** then write in the same email address used to sign up before.
3. Left-click on the **Password textbox** then write in the same password used with the email address written in step 2.
4. Left-click on the **Login button**.

Both the Signup and Login buttons will lead them to the main layout of the program. The login screen layout is found in Figure 1.1.

Next, a user can enter an old chatroom, by LMC on the title of the chatroom within the **Chatroom manager**. A user could then upload a file into the *chatroom*, which allows the AI chatbot to use that file in its response. Then a user can ask a question to InfoGrep using the **Chatroom textbox** at the bottom of the screen. Hit *Enter* after typing out their question, or the **Send Icon** to submit their question to InfoGrep servers. InfoGrep will reply within 5-10 seconds. The *InfoGrep reply* will look akin to Figure 1.2.

2.6 Organization of this Manual

The rest of the manual is organized mainly based on use cases. Chapter 3 includes the Domain mode, Use case diagram, and the Assumptions, Exceptions, and Variances list. Chapter 4 describes the basic use cases affecting the chatroom and the files. What to do if you have trouble and tips on how to use the program better are given in Chapter 5. Chapter 6 gives the limitations of the current version of InfoGrep and the Bibliography.

Chapter 3 - Domain Model, Use Case Diagram, System Architecture Diagram, and Assumptions, Exceptions, and Variances

3.1 Domain Model

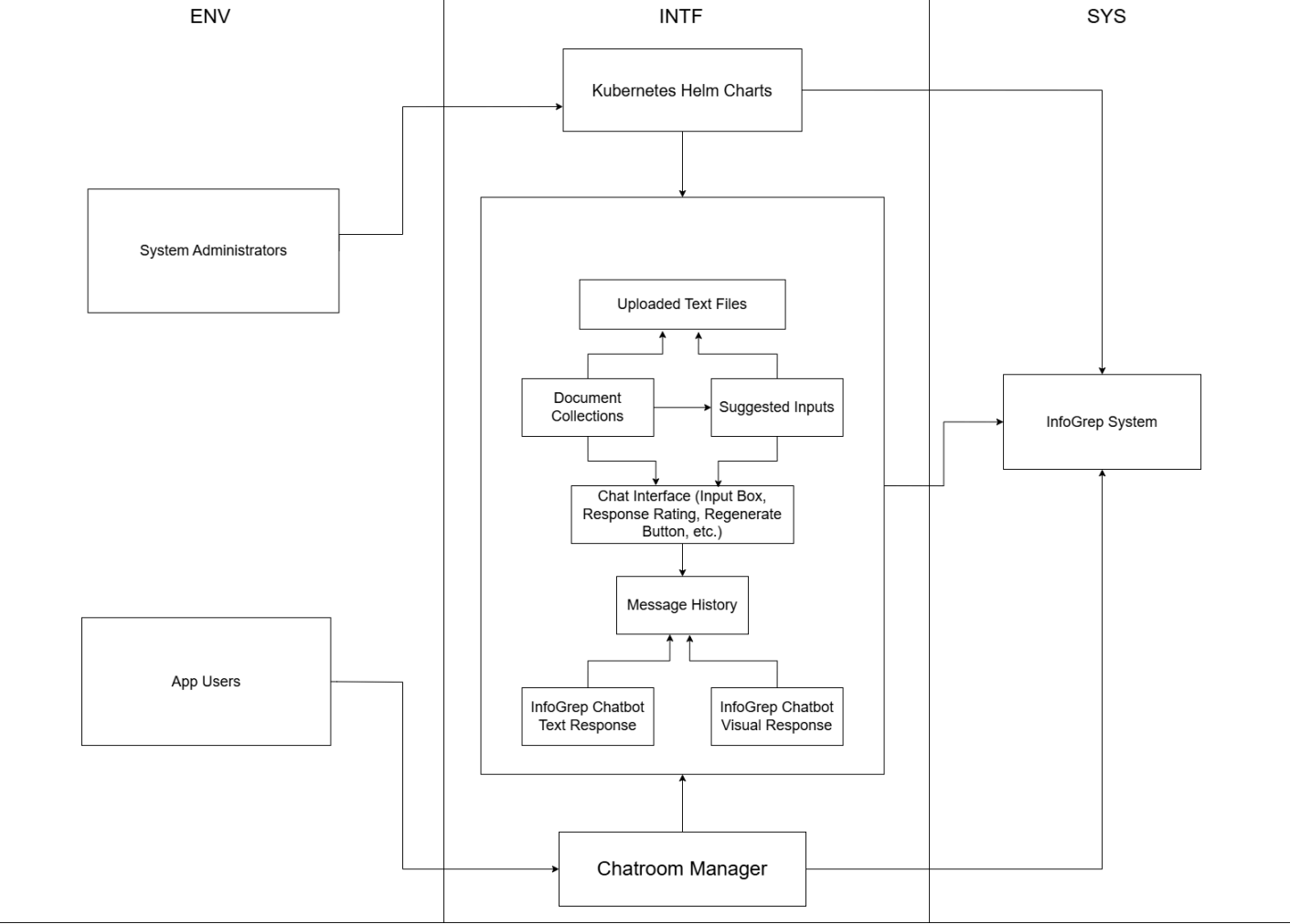


Figure 3.1 Domain Model Diagram

3.2 Use Case Diagram

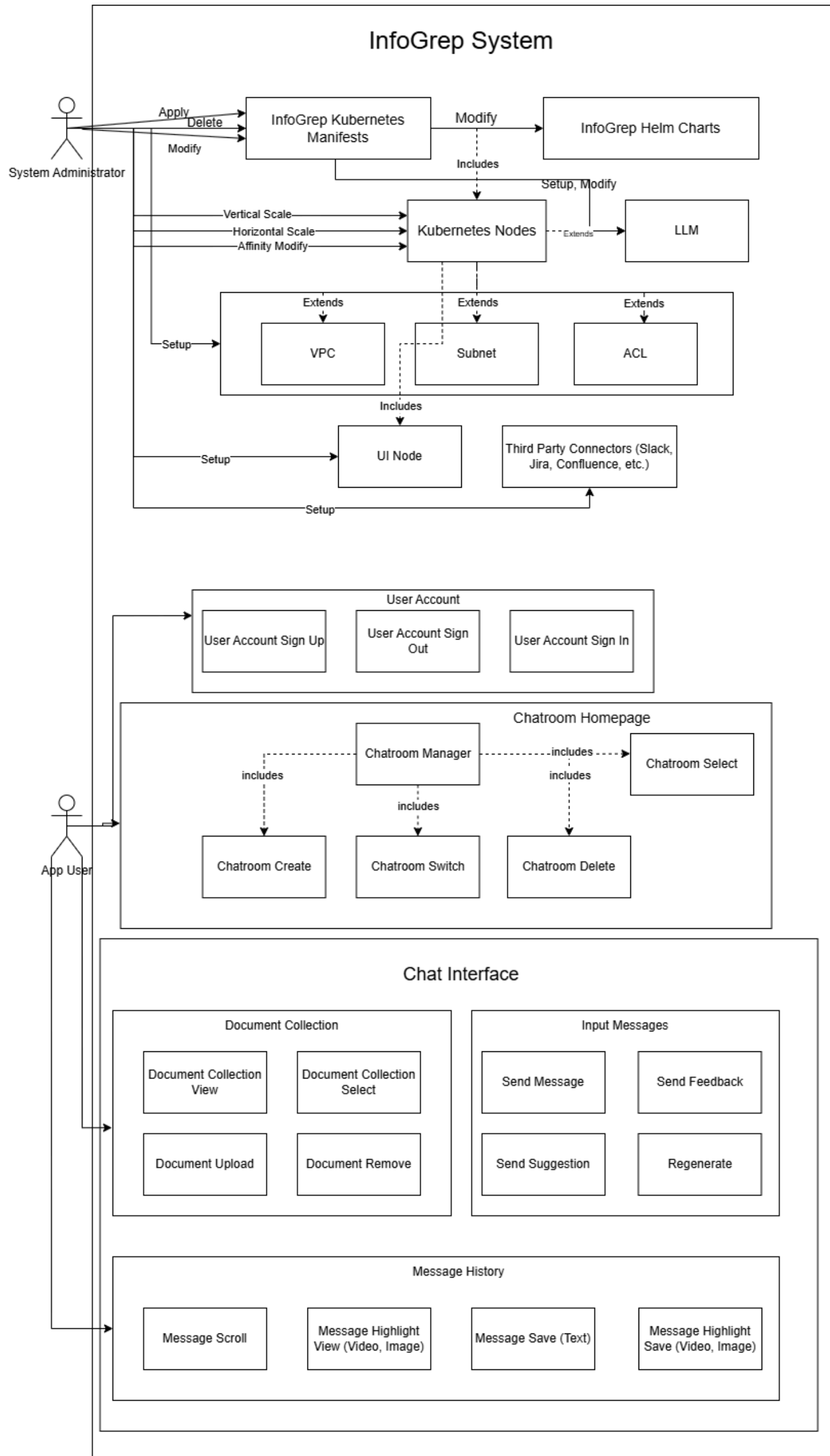


Figure 3.2 Use Case Diagram

3.3 Assumptions, Exceptions, and Variances

3.3.1 SA (System Administrator) Assumption

- There will be one or multiple SA(s) available for setting up the InfoGrep system following the setup instructions.

- The InfoGrep SA (system administrator) is assumed to have advanced knowledge of the InfoGrep user group's software infrastructure including:
 - If operating on the cloud, the SA is assumed to have an advanced knowledge of the details of the cloud provider services. For example, for working with AWS, the SA is assumed to know how *ELB (Elastic Load Balancing)* works and can connect *EKS (Elastic Kubernetes Service)* nodes behind an ELB.
 - If operating on-premise, the SA is assumed to know the architecture of the on-premise infrastructure and can modify and integrate the InfoGrep system.
- The InfoGrep SA is assumed to have a decent knowledge of the following technologies:
 - Docker
 - Kubernetes
 - Helm
 - Computer Networking
- The InfoGrep SA is assumed to have access to on-premise or cloud infrastructure capable of running InfoGrep with CPU & Memory to spare.
- The InfoGrep SA is assumed to have access to on-premise or cloud infrastructure capable of running LLMs with CPU & Memory to spare.
- The InfoGrep SA is assumed to have followed the InfoGrep setup instructions correctly and the infrastructure should be connected and backed by all of the Infogrep microservices at all times.

3.3.2 Application User Assumption

- The application user of InfoGrep is assumed to know the basics of how to operate a computer including:
 - How to boot up your operating system,
 - How to connect to a working network,
 - How to open up and use a web browser,
 - How to connect working input devices, for example, mouse and keyboards, to the computer and interact with the InfoGrep UI
- The user has valid and available emails for registering an InfoGrep account.
- When the user clicks to enter a chatroom, e will enter the correct chatroom.
- The user will submit at least one file before chatting with the chatbots for assistance.
- The user will be able to log into their InfoGrep account with the email and password they used for signing up.
- The user will submit files only in English.
- The user will submit non-corrupted files.
- The user will submit non-empty files with sanitized content.
 - An example of sanitized vs. un-sanitized content could be → File with binary machine code vs File in English about the hydrological cycle taught in Earth123 unit 3.
- When the user submits a third-party document URL (e.g. Jira), the chatbot will be able to access it accordingly and provide a response based on that document URL.
- The user will chat with the chatbot only in English.
- The user will always have access to their chatrooms and document collections.
- The user will always receive a valid response from the chatbot after sending them a message. A valid response should:
 - Contains only context from documents within the current chatroom's document collection.
 - Contains accurate, correct, and unbiased information.
 - Be in English and a human-readable format.
- All media files from **Highlight Items** are always accessible.

- All document collections associated with each chatroom will always be accessible and remain the same unless the user performs an addition or deletion to the collection.

3.4 Exceptions

3.4.1 SA (System Administrator) Exceptions

- There are no system administrators to set up InfoGrep.
- The SA(s) do not have enough knowledge of the InfoGrep user group's software infrastructure including:
 - The SA does not have enough knowledge of the details of the cloud provider if operating on cloud. For example, if the enterprise user group's current infrastructure is in AWS, the SA cannot attach a persistent volume to an EKS container.
 - The SA does not have enough knowledge of the details of the user group's on-premise infrastructure if operated with on-premise infrastructure. For example, the SA does not know how to expose a public URL of InfoGrep for all users to access from their on-premise server.
- The SA(s) do not have a decent knowledge of
 - Docker, for example, e doesn't know how to build a docker image from a Dockerfile.
 - Kubernetes, for example, e doesn't know what a Pod in Kubernetes is.
 - Helm, for example, e doesn't know how to modify values in helm templates to fit their resource needs.
 - Computer networking, for example, e doesn't know how to allocate a private IPv4 block to expose the InfoGrep UI service.
- The enterprise infrastructure cannot run or is not compatible with running InfoGrep.
- The enterprise LLM cannot run or is not compatible with running InfoGrep.
- The enterprise infrastructure runs out of storage.
- The enterprise infrastructure runs out of computing resources such as Memory.
- The enterprise infrastructure gets disconnected from one or all of the microservices.

3.4.2 Application User Exceptions

- The user does not have a basic understanding of how to operate a computer including:
 - How to boot up the operating system
 - How to open up and use a web browser
 - How to connect working input devices, for example, mouse and keyboards, to the computer and interact with the InfoGrep UI.
 - The user has no internet connection
 - The user has an internet connection but cannot connect InfoGrep to it.
- The user has no available email for InfoGrep registration.
- The user clicks and enters a Chatroom and ends up in the wrong Chatroom.
- The user did not upload any documents but started chatting with the chatbot.
- The user forgot about the password or used the correct set of credentials but cannot log into InfoGrep.
- The user submits files in different languages.
- The user submits corrupted files.
- The user submits empty files or files with garbage content.
 - For example, a file with binary machine code.
- The user submits a third-party document URL (e.g. Jira), but InfoGrep does not have access to it.
- The user attempts to chat in an unsupported language.
- The user is not able to access their chatrooms and document collection.

- The user entered the chatroom and tried to send messages but the chatbot did not respond.
- The chatbot returns irrelevant content from the user query.
- The chatbot returns sensitive content.
- The chatbot returns inaccurate content.
- The chatbot returns content from a wrong document collection as opposed to the document collection the user is querying against.
- The chatbot does not return a message.
- The media file returned by the chatbot is not accessible.
- The document collection associated with a chatroom is not accessible or incorrect.

3.5 Variations

- The setup files are modified by the system administrators.
- The user has an internet connection, but the speed or bandwidth makes it impossible for them to receive data from InfoGrep.
- The user registered for an account that already exists in the InfoGrep database.
- The user registered with a password that does not meet the InfoGrep password requirement.
 - Minimum/Maximum length,
 - Syntax requirements, (numbers, special characters, capital letters, etc.)
 - Contains username or personal information in it.
- The LLM models were switched in the middle of generating content for some users.
- The microservices started an upgrade in the middle of communicating with InfoGrep due to user actions such as
 - Account sign-up or sign-in
 - Retrieving user chatroom list
 - Retrieving chatroom documentation lists
 - Handling user file upload, chatroom creation, collection creation, etc.
- The user submitted a file with the same name and content as an existing file in the document collection.
- The user submitted a file with the same content but with different names.
- The user submitted a file with the same name but with different content.
- The user submitted the file, the file is in the middle of getting uploaded to the system, and
 - The internet got disconnected,
 - The InfoGrep infrastructure went down,
 - The user's device got powered off.
- The user submits a file but deletes it before the upload is completed.
- The user has a valid file, but their current operating system cannot open the file.
- The user logged in from a new device, with the old device not logged out.
- The user sent the chatbot messages before getting a response from the last message.
- The user added or removed a file from the current chatroom's document collection after sending the chatbot a message but before receiving a response.
- The user deleted the current chatroom after sending the chatbot a message but before getting a response back.
- The user changed to a different chatroom after sending the chatbot a message but before getting a response back.
- The user sent the chatbot a message regarding previously generated content in the chat history (before the last message).
- The user uploaded a file with conflicting information to the same document collection.

- The user uploaded documents with a clear bias.
- Document provider failure (e.g. Slack), several variations for this sublist item
 - The access token for the Slack API expired,
 - The response format from Slack changed,
 - Slack went down while InfoGrep is querying it,
 - InfoGrep got rate-limited by Slack while a chatbot was generating the response.
- The user sends the chatbot a message, where the message is ambiguous or contains a typo(s) (e.g. Explain geology of Austria → Explain geology of Australia).
- The user asked the chatbot a question in slang or jargon that newly came out. (Explain the concept of concurrency with no cap).
- The user uploaded an extremely large file (i.e. 100G).
- The chatbot replied in a format that does not follow markdown. (e.g. it displays `###Title` instead of a heading).

3.6 Exceptional Scenarios and Alternative Scenarios Prevention and Handling for System Administrator

3.6.1 Human-Level Exceptional Scenarios Prevention and Handling

- In cases where there are no system administrators to set up InfoGrep, the InfoGrep system will ignore this exception and the system will not be available or functional. No application user will be able to access InfoGrep.
- If operating on cloud, in cases where the SA does not have enough knowledge of the cloud provider, the setup might be incomplete or incorrect, which could cause the system to be dysfunctional or unreachable. InfoGrep will not be able to provide direct help in this case, but there will be links to the official cloud provider documents in the official InfoGrep GitHub [README.md](#) that the SA could visit for external help. For example, the official [AWS EKS documentation](#) will be linking the InfoGrep README.md.
- If operating on-premise, in cases where the SA does not have enough knowledge of the enterprise's infrastructure, the setup might be incomplete or incorrect, which could cause the system to be dysfunctional or unreachable. InfoGrep will not be able to provide any help in this case and ignore this exception, and the system will be unavailable until the SA has enough knowledge to set up InfoGrep correctly.
- If the SA does not have a decent knowledge of Docker, InfoGrep will not be able to provide any direct help or knowledge regarding Docker to the SA. InfoGrep will provide commands related to Docker in the setup instruction that the SA could look at for setup, but if the SA does not know how to install Docker or how to correctly pull an image from a Docker registry, then InfoGrep will be unavailable, and ignore this exception until the SA knows how to do so.
- If the SA does not have a decent knowledge of Kubernetes and Helm, InfoGrep will not be able to provide any direct help or knowledge regarding Kubernetes and Helm to the SA. InfoGrep will provide commands related to Kubernetes and Helm in the setup instruction, the SA could use those for setup, but if the SA does not know how to install Helm, or does not have a correctly configured Kubernetes cluster that is reachable, InfoGrep will be unavailable and ignore this exception until the SA knows how to do so. InfoGrep will also provide a link to the [official Helm documentation](#) in the setup instructions as a reference for the SA.
- If the SA does not have a decent knowledge of computer networking, InfoGrep will provide a connection troubleshooting guide that is included in the NOTES section in the official InfoGrep GitHub repository. The SA may or may not find the answers they seek with their specific networking problem there, in cases where they can't find the answer they seek, InfoGrep will not be able to continue providing any direct knowledge or help regarding their network issue and ignore this exception. The InfoGrep system will not be available until the SA gains enough knowledge on this topic.

3.6.2 Infrastructure-Level Exceptional Scenarios Prevention and Handling

- If the enterprise infrastructure cannot run or is not compatible with InfoGrep, InfoGrep will not be able to provide an alternative solution or help regarding the incompatibility. There will be notes on the requirement of running InfoGrep both in the InfoGrep [README.md](#) as well as the setup instruction to try to prevent this exception from happening, but ultimately, there is no way InfoGrep could block this exception from happening. InfoGrep will not be available if there is no compatible infrastructure.

- If the enterprise LLM cannot run or is not compatible with InfoGrep, there will be documents included in the InfoGrep LLM setup page (WIP) to prevent this exception from happening, but ultimately this does not block this exception from happening. InfoGrep provides a solution to this exception by providing some InfoGrep public LLMs that are compatible with the framework, the SA could set the system up with the InfoGrep-provided LLM for it to be functional.
- If the enterprise infrastructure runs out of storage, the InfoGrep system will still be reachable and available to the existing application users. However, users will not be able to upload more documents, and new users will not be able to sign up. InfoGrep will provide a scaling mechanism with Helm and Kubernetes that the SA could apply to their infrastructure once more storage unit is added to the system. InfoGrep will not be able to resolve this exception until a new storage unit is added.
- If the enterprise infrastructure runs out of computing resources such as memory or CPU, InfoGrep provides resource adjustment and allocation mechanisms through Helm and Kubernetes. If the computing resource in total does not meet the requirement of running InfoGrep, for example, there are only 1GB of RAM and 0.5 CPU cores available, then InfoGrep will not be able to resolve this exception and the system will be unavailable until more resources are introduced.
- If the enterprise infrastructure gets disconnected from one or all of the microservices, InfoGrep provides a retry mechanism through Kubernetes networking as well as a retry mechanism implemented in each microservice to try to reconnect to the cluster. InfoGrep also provides disaster recovery and replica sets for each microservice through Kubernetes to ensure the availability of each microservice. In extreme situations, such as the entire AWS goes down or if running on-premise, the entire company building goes out of power, InfoGrep will not be available or reachable until AWS comes back online or the company building restores power.

3.6.3 Human-Level Alternative Scenarios Prevention and Handling

- If the setup files are modified by the SA(s), InfoGrep will ignore this variation. This could either have no impact on the system setup or could cause the system to be unavailable or not accessible by the application users depending on the SA(s)'s changes.

3.7 Exceptional Scenarios and Alternative Scenarios Prevention and Handling for Application User

3.7.1 Human-Level Exceptional Scenarios Prevention and Handling

- If the user does not have a basic understanding of how to operate a computer, for example, the user does not know
 - how to boot up er computer and operating system
 - how to open up and use a web browser
 - how to connect their computer with a working input device
 - has no working internet or does not know how to connect their computer to a working internet

Infogrep will not be able to handle these exceptions and will ignore them. Infogrep will not be available or usable.

- If the user does not have an available email for InfoGrep registration, InfoGrep will not be able to handle this exception and will ignore it. InfoGrep will still be reachable in this case and the user will be able to access the **Login Page**. However, the user will not be able to access the actual **Chatroom Manager** or chat with the chatbots.

3.7.2 System-Level Exceptional Scenarios Prevention and Handling

- InfoGrep will implement a database system that stores a mapping of user accounts with their corresponding chatrooms and internal chatroom routers to ensure the users access the correct chatrooms they click.
- If the user did not upload any documents but started chatting with the chatbot, InfoGrep will prevent the user's message from sending and prompt them with a message to upload some valid documents before chatting with the chatbot.
- InfoGrep will implement a database system that stores username and their corresponding encrypted password for verifying user login and to ensure the validity when a user tries to sign in, in order to prevent the mismatching username and password exception from happening.
- If the user forgets the login username or password they used for signing up for an InfoGrep account, InfoGrep will implement an account recovery system. Users will be able to click the forgot my user name button or forgot my

password button on the login page, and proceed with verifying their emails by entering the verification code InfoGrep sent to them and resetting their password or recovering their username.

- If these cases where the user uploaded an invalid document such as:
 - A file not in English, the InfoGrep file parser will throw an `InvalidContent` error when the file starts parsing, and notifies the user of an unsuccessful document upload. The document will not be uploaded to the document collection.
 - An empty file, the InfoGrep file parser will throw an `EmptyFile` error when the file starts parsing and notifies the user of an unsuccessful document upload. The document will not be uploaded to the document collection.
 - Corrupted files or files with garbage content such as binary machine code, the file parser will throw an `InvalidContent` error when the file starts parsing, and notifies the user of an unsuccessful document upload. The document will not be uploaded to the document collection.
- If the user submits a third-party document URL (e.g. Jira), but InfoGrep does not have access to it, the InfoGrep parser will throw a `NotAuthroized` error when the file/link starts parsing and notifies the user of an unsuccessful document upload. The document/link will not be uploaded to the document collection.
- If the user attempts to chat in some language other than English, InfoGrep will analyze the user question after the user message is sent and notify the user of an unrecognized language, and prompt them to try again. The user question will not reach the actual generation system and the chatbot will not generate a response.
- InfoGrep will be highly available and accessible with mechanisms implemented with Kubernetes Deployment, Statefulset, and Loadbalancer. This ensures that chatrooms and document collections will always be available and accessible to users with no downtime.
- InfoGrep will implement a 1 to 1 channel between chatrooms and users, the InfoGrep will detect any user input through the channel and starts a generation process. A message will be ensured to send to the users through a few try-catch blocks, so even if the generation was not successful, the user will still receive a message back indicating the generation failed, thus ensuring that every user message has a corresponding chatbot response.
- InfoGrep chatbot response will only be generated from documents uploaded to the currently selected document collection to prevent irrelevant content in the generated response.
- InfoGrep has an internal list of sensitive words that will get applied before the generated content gets returned to the user. Any sensitive word will get filtered out and replaced with a `<sensored>` replacement for the actual generation to prevent sensitive content in the generated response.
- InfoGrep currently ignores the exception where the generated content might be inaccurate. Although the system does not prevent this exception from happening, InfoGrep will include the source of the original document along with the generated content for the user to verify the accuracy themselves.
- InfoGrep implements a database system that stores all document collections stored in a chatroom, and all documents stored in a document collection. This relation ensures the user will query against the correct documents by checking their current selected document collection and chatroom.
- InfoGrep implements persistent volume storage with Kubernetes PersistentVolume and PersistentVolumeClaim to ensure all media files stored in a chatbot response will always be accessible to users.

3.7.3 Human-Level Alternative Scenarios Prevention and Handling

- If the user has an internet connection, or has enough operational knowledge with their computer such as booting up their operating system, or using a web browser to interact with InfoGrep, but is limited by external hardware limitations such as speed or bandwidth of their network, or their personal computer does not have enough computing resource such as CPU or RAM for them to interact with InfoGrep. In this case, InfoGrep will ignore this variation. This could cause InfoGrep to be either
 - Available but unstable, disconnects sometimes, or the user's computer crashes,
 - Unavailable.

3.7.4 System-Level Alternative Scenarios Prevention and Handling

- InfoGrep has implemented a database system that stores all registered usernames. InfoGrep will conduct a database check for duplicated usernames if the user tries to register under an already existing username. InfoGrep will send the user an error message and prompt er to try register again with a different username to prevent a duplicated user account.

- If the user attempts to register with a password that does not meet the InfoGrep password requirement of:
 - Minimum/Maximum Length
 - Syntax requirements, (numbers, special characters, capital letters, etc.)
 - Contains username or personal information in it.

InfoGrep will block the user from registering and prompt them to try again with a different password since they did not meet the password requirement. This is done through a Regex expression matching in the InfoGrep User Service.

- If the user LLMs were switched in the middle of a generation for some users, the InfoGrep AI Service will stop its current generation and erase the current generation memory. It will then post a generation request to itself and start a new generation with the new LLM.
- If the microservice started an upgrade in the middle of communicating with InfoGrep due to user actions such as
 - Account sign-up or sign-in
 - Retrieving user chatroom list
 - Retrieving chatroom documentation lists
 - Handling user file upload, chatroom creation, collection creation, etc.

InfoGrep has implemented a ReplicaSet system with Kubernetes, where each service is backed up by two or more containers. A rolling update strategy is also implanted by InfoGrep with Kubernetes, a microservice will always have a replica available for taking in requests while the other replica(s) is upgrading to prevent any downtime for application users.

- If the user submits a file with the same name and content as an existing file in the document collection, InfoGrep will prompt the user whenever the user tries to upload the same file. This is implemented with the InfoGrep File Management Service with a database that stores the documents in the user's document collection that checks for duplicated files for an upload.
- If the user submits a file with the same content but a different name, InfoGrep will allow this document to be uploaded. In the InfoGrep Vector Management Service, each parsed chunk of text from the original document will contain a mapping of the original text and the vector embedding in the vector database. To ensure fairness in the eventual generation, InfoGrep has implemented an algorithm that checks and eliminates any duplicated vector retrieval results from the topN candidates and refills the topN results with the next available candidate until it has N results with different original texts ready.
- If the user submits a file with the same name but different content, the InfoGrep file duplication check mentioned earlier in this section will fail and InfoGrep will think the user is trying to upload an existing file into the document collection again. The user will be forced to change the name of the file in order to upload this document.
- If the user submitted the file, the file is in the middle of getting uploaded to the system, and
 - The internet got disconnected
 - The InfoGrep infrastructure went down,
 - The user's device got powered off.

The user's file will not get successfully uploaded and InfoGrep will stop the file upload process in the File Management Service to ensure data correctness.

- If the user deletes a file before the upload is completed, the upload will be canceled and InfoGrep will stop the current upload process in the File Management Service to ensure data correctness.
- If the user is trying to upload a valid file, but er operating system is not compatible with opening the file type, InfoGrep will not be able to provide any remediation and will ignore this variation. The user will not be able to upload this file until e finds an operating system that is compatible with opening the file.
- If the user logs in from a new device, with the old device not logged out, InfoGrep will automatically log the old device out. This is implemented through the InfoGrep User Service, where each user is assigned an active session while they are logged in. Once a new login is detected, the old session will expire and be deleted from the User Service, and this will trigger the log-out from the old device.
- If the user sends the chatbot messages before getting a response from the last message, InfoGrep will provide a corresponding response to each message sent by the user. This is implemented by the InfoGrep Chatroom Service

where there is a 1 to 1 channel established for all user messages. Each channel will remember the user message ID and store it in a temporary cache and when the generation is completed, a response will find its corresponding message ID and get sent to the user, when this transaction is done, this message ID will get deleted from the cache to ensure each message will always have a corresponding response.

- If the user sends the chatbot a message regarding previously generated content in the chat history (before the last message), the InfoGrep chatbot will provide an approximate response along with the context of the previously generated content. This is implemented in the InfoGrep Chatroom Service where the service will keep a summarized memory of the chat history between each document collection and user messages.
- If the user uploads a file with conflicting information to the same document collection or the document has a clear bias. InfoGrep will not be able to handle the variation where these variations and cannot guarantee the generated response's accuracy and fairness. As mentioned in section 3.7.2, InfoGrep will provide the source of the document along with the generated content so the application user can verify the accuracy and fairness of the generated content.
- If the document provider (e.g. Slack):
 - The access token for the Slack API expired,
 - The response format from Slack changed,
 - Slack went down while InfoGrep is querying it,
 - InfoGrep got rate-limited by Slack while a chatbot was generating the response.

InfoGrep will return a `ServiceAccessError` error along with a generation failure message to the user. The generation will not be successful and InfoGrep will not save this generation into the Chatroom memory.

- If the user sends the chatbot a message, where the message is ambiguous or contains a typo(s) (e.g. Explain geology of Austria → Explain geology of Australia), or the user asks the chatbot a question in slang or jargon that newly came out. (Explain the concept of concurrency with no cap). InfoGrep will not be able to identify and prevent these variations, they will be ignored. The generated content in this case might be inaccurate or biased. As mentioned in section 3.7.2, InfoGrep will provide the source of the document along with the generated content so the application user can verify the accuracy and fairness of the generated content.
- If the user uploads an extremely large file (i.e. 100G), InfoGrep has set a default file size limit of 2GB for file uploads and thus the 100G file will not get uploaded. InfoGrep will prompt the user of an invalid file size and the user will not be able to upload the 100G file. This limit could be modified by the SA by modifying values in the Helm template `values.yaml`
- InfoGrep chatbot's output will be implemented by the react-markdown library to ensure all outputs generated follow the correct markdown format.

3.8 System Architecture Diagram

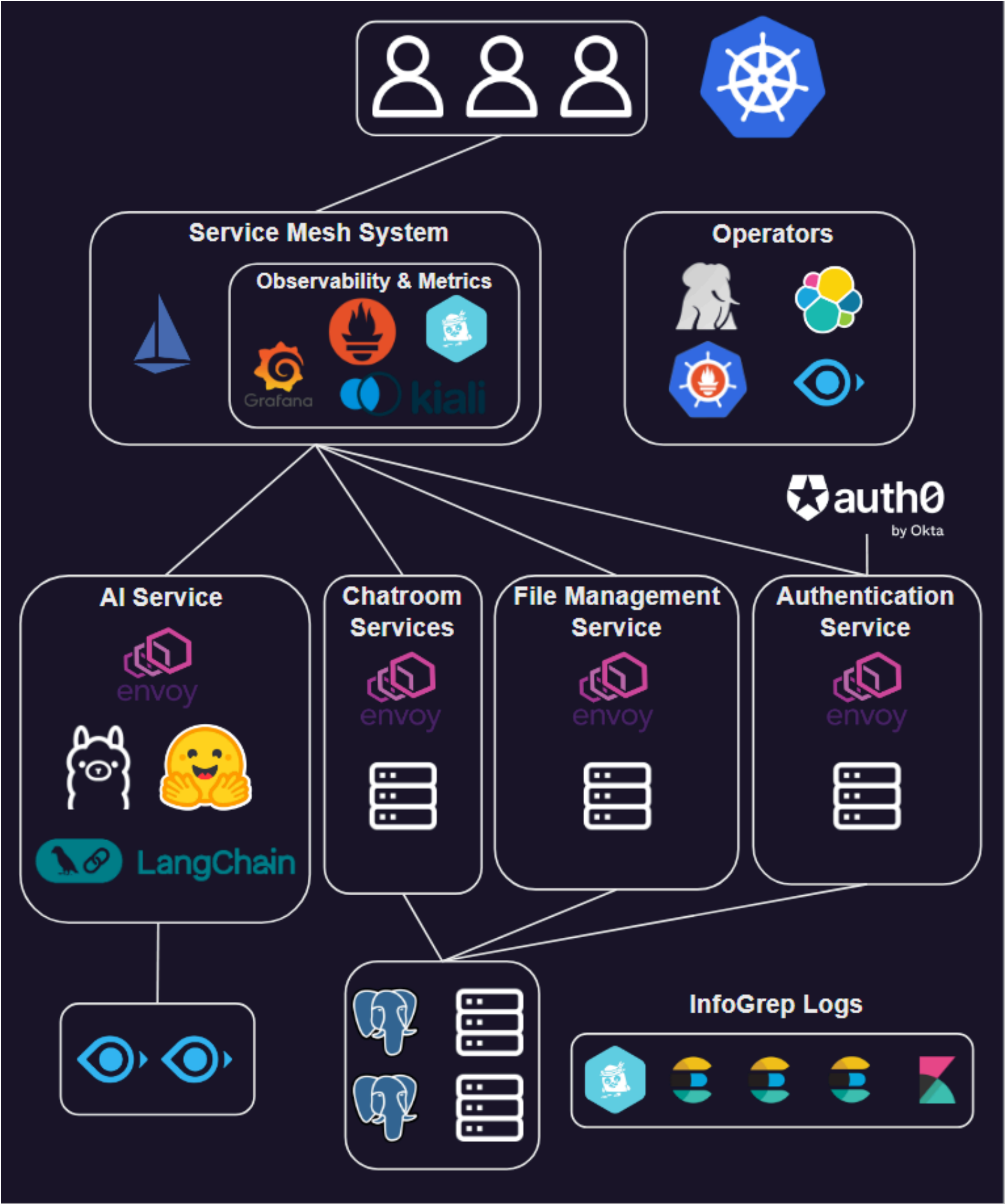


Figure 3.8 System Architecture Diagram

Chapter 4 - Basic Use Cases

Basic use cases are classified according to how they affect the two user groups of InfoGrep — the application users, and the SAs.

4.1 InfoGrep Configuration

4.1.1 InfoGrep Helm Charts and Default Value

Below is the high-level InfoGrep Helm Charts definition.

```
apiVersion: v2
name: infogrep
description: Helm Chart for InfoGrep Kubernetes Manifests
type: application
version: 0.1.0
appVersion: "0.1.0"
maintainers:
- email: infogrep.uw@gmail.com
  name: infogrep
```

All infrastructure level values are included in the `values.yaml` file in the `infogrep-charts` directory.

The SAs will be able to override the default value defined in the `values.yaml` file by creating a `config.yaml` from the provided `config.template.yaml` with a command such as `cp config.template.yaml config.yaml` .

Below is the content for `config.template.yaml`

```
fullNameOverride: ""
deploymentType: dev # dev | local | cloud
replicaCount: 1
global:
  enableMetrics: false

Ingress:
  className: "nginx" # nginx | traefik-v2 | traefik-v3 | istio-gateway
  host: "local.infogrep.ai"

# -- Ai Service overrides
AiService:
  ollamaCreate: false
  env:
    openaiKey: "placeholder"
    ollamaUrl: "host.minikube.internal:11434"
  service:
    type: ClusterIP
  image:
    tag: latest
  resources:
    limits:
      cpu: 500m
      memory: 1Gi
    requests:
      cpu: 500m
      memory: 512Mi

# -- Auth Service overrides
AuthService:
  env:
    CLIENT_ID: ""
    CLIENT_SECRET: ""
    DOMAIN: ""
    APP_SECRET_KEY: ""
    REDIRECT_URI: ""
    FRONTEND_LOGIN_URI: ""
  service:
    type: ClusterIP
  image:
    tag: latest
```



```
resources:
  limits:
    cpu: 500m
    memory: 512Mi
  requests:
    cpu: 500m
    memory: 512Mi

# -- Chatroom Service overrides
ChatroomService:
  service:
    type: ClusterIP
  image:
    tag: latest
  resources:
    limits:
      cpu: 500m
      memory: 512Mi
    requests:
      cpu: 500m
      memory: 512Mi

# -- File Management Service overrides
FileManagementService:
  service:
    type: ClusterIP
  image:
    tag: latest
  resources:
    limits:
      cpu: 500m
      memory: 512Mi
    requests:
      cpu: 500m
      memory: 512Mi

# -- UI service overrides
InfogrepFrontend:
  create: true
  service:
    type: LoadBalancer
  image:
    tag: latest
  resources:
    limits:
      cpu: 500m
      memory: 2Gi
    requests:
      cpu: 500m
      memory: 1.5Gi

# -- Milvus overrides
MilvusConfig:
  serviceType: ClusterIP
  standaloneReplica: 1
  env:
    username: "infogrep"
    password: "placeholder"
```

```

    rootPassword: "placeholder"
  persistence:
    storageClass: ""
    size: 5Gi

  MilvusEtcdConfig:
    replicaCount: 1
    persistence:
      storageClass: ""
      size: 5Gi

# -- Elasticsearch overrides
  ElasticsearchConfig:
    replicaCount: 1
    defaultPodSpec: true
    persistence:
      storageClass: ""
      size: 3Gi

# -- Kibana overrides
  KibanaConfig:
    replicaCount: 1
    serviceType: "NodePort"
    defaultPodSpec: true

  InfogrepPostgres:
    replicaCount: 2
    postgresService:
      ro-enabled: true
      r-enabled: true
    env:
      username: "postgres"
      password: "placeholder"
    persistence:
      storageClass: ""
      size: 3Gi

```

4.1.2 Updating Infogrep Configurations

Configuring Global Values

There are a couple of global variables to configure of an Infogrep deployment. For example:

- Horizontal scaling of all Infogrep managed services could be done by changing the `replicaCount` global field, the default value for `replicaCount` is one, which means each Infogrep service will only have one replica running for handling traffic.
- The SA can modify the deployment type by modifying the `deploymentType` global field, by making it either `dev`, `local`, or `cloud`, which Infogrep will adjust accordingly to spawn resources for the specified deployment type, for example, a public load balancer address of the Infogrep UI for a `cloud` deployment.
- The SA could also choose to enable or disable the observability dashboards the Infogrep provides by setting the `global.enableMetrics` field to either `true` or `false`. If this field is set to `true`, Infogrep will deploy additional resources such as Grafana Dashboards and Prometheus metrics scraper as a part of the deployment.
- The `Ingress.className` and `Ingress.host` field specifies what type of Ingress controller to use and the host URL that Infogrep backend services will be exposed at.

Configuring an Infogrep Service

Using the Authentication Service values in the `config.template.yaml` shown in the previous section as our example here.

- The SA could perform a vertical scale and modify the resource they wish to use for the Authentication Service deployment by modifying the resource allocation configs here. In this example, we configured the AuthService to

request for half of a logical CPU core and 512Mi of memory. The AuthService will also have a hard resource limit of 1 logical CPU core and 1Gi of memory. In case where the application needs more resources than the configured limit to run, the current AuthService will halt itself until more resources are provisioned.

```
resources:
  limits:
    cpu: 1000m
    memory: 1Gi
  requests:
    cpu: 500m
    memory: 512Mi
```

- The SA could set the environment variables that the AuthService needs by configuring the `env` fields. A sample OAuth configuration is shown below.

```
env:
  CLIENT_ID: "aabbccddeee"
  CLIENT_SECRET: "client-secret"
  DOMAIN: "test123.ca.auth0.com"
  APP_SECRET_KEY: "12334"
  REDIRECT_URI: "http://api.infogrep.ca/auth/authorize"
  FRONTEND_LOGIN_URI: "http://app.infogrep.ca"
```

- The SA could specify a version of the AuthService to deploy by modifying the `image.tag` field. In this example, we configured the AuthService to use version 0.1.2

```
image:
  tag: 0.1.2
```

- The SA could configure the visibility of the service by modifying the `service.type` field. In this example, we used `ClusterIP` as the service type, which indicates the service will not be public accessible and will only be available for accessing inside of the cluster.

```
service:
  type: ClusterIP
```

- The SA could configure the storage provisioner and size desired for a stateful service in the respective section within the `config.template.yaml` . In this example, we configured the Infogrep PostgreSQL database cluster to have 2 nodes, and we are using the Longhorn storage provisioner and giving each node 3Gi of space for storing data.

```
InfogrepPostgres:
  replicaCount: 2
  postgresService:
    ro-enabled: true
    r-enabled: true
  env:
    username: "postgres"
    password: "placeholder"
  persistence:
    storageClass: "longhorn"
    size: 3Gi
```

4.2 InfoGrep Deployment

4.2.1 InfoGrep Deployment Command

To deploy Infogrep, navigate to the root Infogrep directory and run `bash scripts/install.sh` .

As shown in the previous section, it is important that the `config.yaml` file is created and configured with appropriate values desired for an Infogrep deployment in order for the setup to be successful.

The setup script will read provided configuration values in `config.yaml` and deploy Infogrep correspondingly, for example, deploy extra resources when the `global.enableMetrics` field is set to true. After executing the command, no additional steps needs to be done by the SA. The setup process for will approximately 5 to 8 minutes.

4.2.2 InfoGrep Deployment Verification

After executing the setup script, the SA should monitor the deployment progress, as an example, run `watch kubectl get deploy -n infogrep` to get periodic updates from the infogrep namespace.

A successful deployment will eventually reach a steady state, where all workload components are available and ready as shown below. Note that this is a sample Infogrep deployment and not all deployment will contain the exact same set of resources depending on how the SA configured Infogrep. Key things to note here is that all pod status should be `Running`, all deployment and stateful sets' ready status should be `1/1`.

NAME	READY	STATUS	RESTARTS	AGE
pod/infogrep-ai-service-5b65998c49-p658z		2/2	Running	2 (6m43s ago) 9m11s
pod/infogrep-auth-service-7ffdcc4fc-749cn		2/2	Running	0 9m11s
pod/infogrep-chatroom-service-64966995cc-zpjsg		2/2	Running	0 9m11s
pod/infogrep-elasticsearch-logs-es-default-0		2/2	Running	0 9m11s
pod/infogrep-file-management-service-f4b85b85d-77zqf		2/2	Running	0 9m11s
pod/infogrep-kibana-kb-5894d94c89-7xc7x		2/2	Running	0 9m10s
pod/infogrep-postgres-1	2/2	Running	0	7m13s
pod/infogrep-postgres-2	2/2	Running	0	113s
pod/infogrep-vectordb-etcd-0		2/2	Running	0 9m11s
pod/infogrep-vectordb-milvus-standalone-84f9978854-vfvs8		2/2	Running	0 6m41s
pod/infogrep-vectordb-minio-7c96f6f69d-j7lfz		2/2	Running	0 9m11s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/infogrep-ai-service	1/1	1	1	9m11s
deployment.apps/infogrep-auth-service	1/1	1	1	9m11s
deployment.apps/infogrep-chatroom-service	1/1	1	1	9m11s
deployment.apps/infogrep-file-management-service	1/1	1	1	9m11s
deployment.apps/infogrep-kibana-kb	1/1	1	1	9m10s
deployment.apps/infogrep-vectordb-milvus-standalone	1/1	1	1	6m41s
deployment.apps/infogrep-vectordb-minio	1/1	1	1	9m11s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/infogrep-ai-service-5b65998c49	1	1	1	9m11s
replicaset.apps/infogrep-auth-service-7ffdcc4fc	1	1	1	9m11s
replicaset.apps/infogrep-chatroom-service-64966995cc	1	1	1	9m11s
replicaset.apps/infogrep-file-management-service-f4b85b85d	1	1	1	9m11s
replicaset.apps/infogrep-kibana-kb-5894d94c89	1	1	1	9m10s
replicaset.apps/infogrep-vectordb-milvus-standalone-84f9978854	1	1	1	6m41s
replicaset.apps/infogrep-vectordb-minio-7c96f6f69d	1	1	1	9m11s

NAME	READY	AGE
statefulset.apps/infogrep-elasticsearch-logs-es-default	1/1	9m11s
statefulset.apps/infogrep-vectordb-etcd	1/1	9m11s

To verify that Infogrep is correctly installed and up and running, the SA should try to access the Infogrep services. To do so, run `kubectl get ingress -n infogrep` to list the ingress deployed in the infogrep namespace. The output should be similar to the following, note that this is only a sample output and not all output is exactly the same depending on how the SA configured the Ingress.

NAME	CLASS	HOSTS	ADDRESS	PORTS	AGE
------	-------	-------	---------	-------	-----

```
infogrep-ingress  nginx  local.infogrep.ai 192.168.49.2 80 17m
```

As shown in the output, Infogrep is currently running at `192.168.49.2` and configured with host `local.infogrep.ai` .

Using the AI Service as an example for testing, the SA could run `curl local.infogrep.ai/ai/api/docs` to try to access the API documentation page for the AI service. A correct output should look similar to the output shown below.

```
<!DOCTYPE html>
<html>
<head>
<link type="text/css" rel="stylesheet" href="https://cdn.jsdelivr.net/npm/swagger-ui-dist@5/swagger-ui.css">
<link rel="shortcut icon" href="https://fastapi.tiangolo.com/img/favicon.png">
<title>AI API Doc</title>
</head>
<body>
<div id="swagger-ui">
</div>
<script src="https://cdn.jsdelivr.net/npm/swagger-ui-dist@5/swagger-ui-bundle.js"></script>
<!-- `SwaggerUIBundle` is now available on the page →
<script>
const ui = SwaggerUIBundle({
  url: '/ai/openapi.json',
  "dom_id": "#swagger-ui",
  "layout": "BaseLayout",
  "deepLinking": true,
  "showExtensions": true,
  "showCommonExtensions": true,

  presets: [
    SwaggerUIBundle.presets.apis,
    SwaggerUIBundle.SwaggerUIStandalonePreset
  ],
})
</script>
</body>
</html>
%
```

The SA could also try to directly visit the url from a web browser. A correctly installed Infogrep deployment should display the following when visiting `infogrep-host/ai/api/docs` .

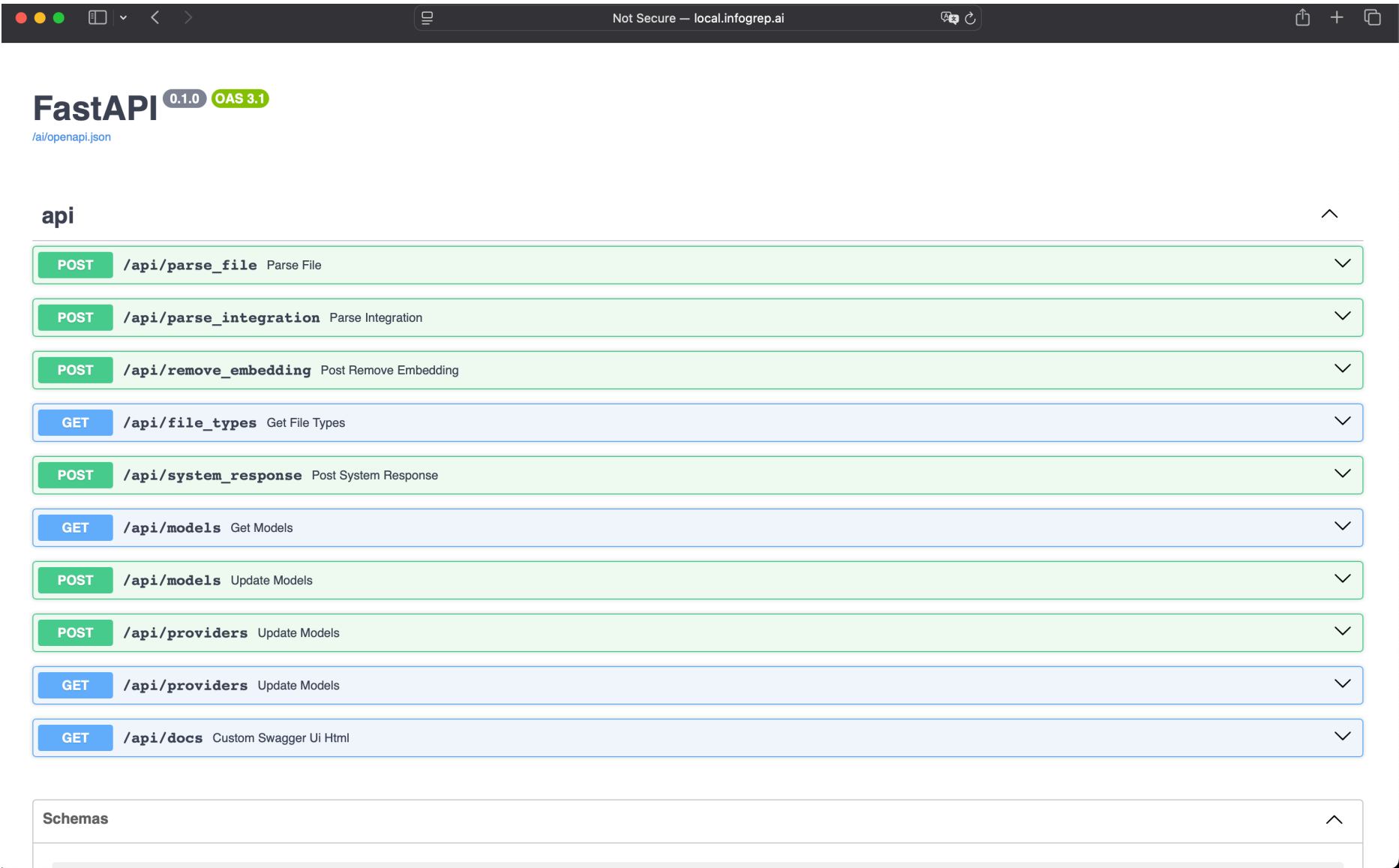


Figure 4.0.1 Infogrep Docs Example

4.2.3 Uninstalling Infogrep

To uninstall Infogrep, navigate to the root Infogrep directory and run `bash scripts/uninstall.sh` .

The script will remove all resource created by Infogrep and remove all variable definitions from the cluster. To verify that Infogrep is successfully deleted, the SA could run `kubect! get ns infogrep` and should see the following output.

```
Error from server (NotFound): namespaces "infogrep" not found
```

4.3 User Account

4.3.1 The User Login Page

The **User Login Page** is the landing page of InfoGrep when the user tries to access the application for the first time. This page allows the user to either sign in via their Business’s SSO account with the **Single Sign On Button** or sign in to an existing InfoGrep account by inputting er account information in the **Email Input Box** and **Password Input Box** and clicking the **Login Button**.

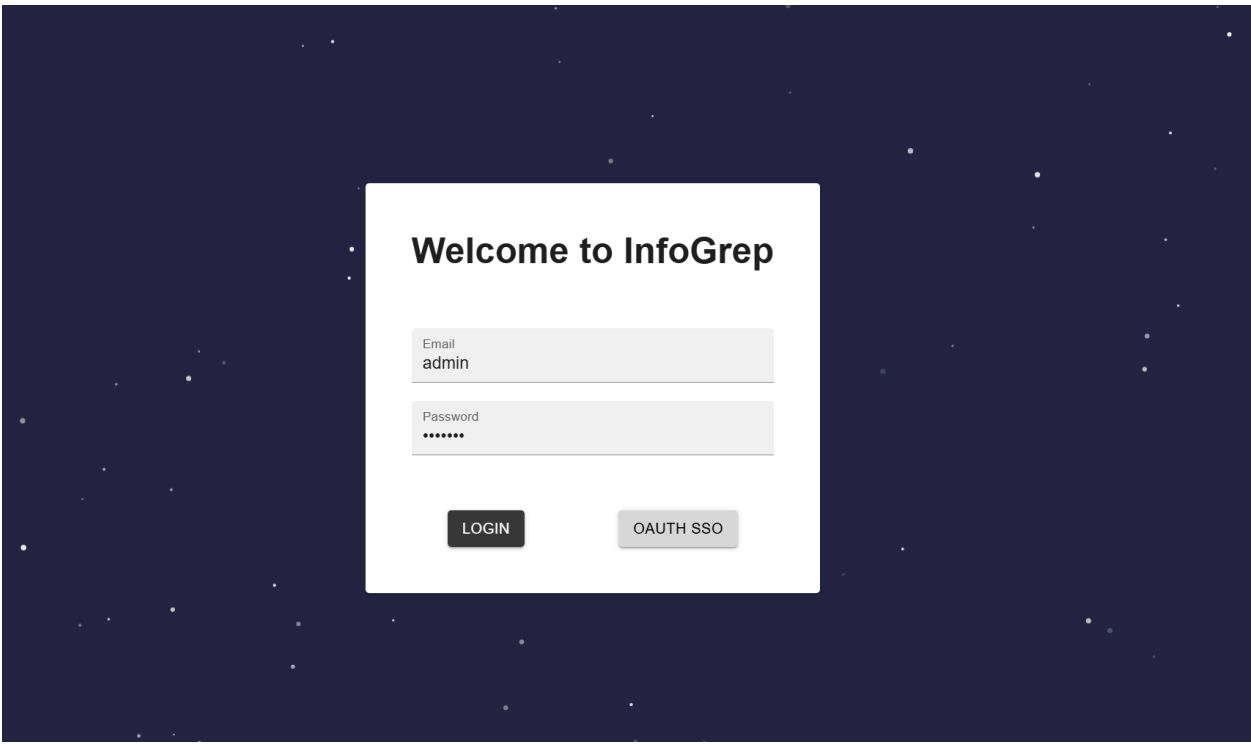


Figure 4.1 The User Login Page

4.3.2 The User Settings Page

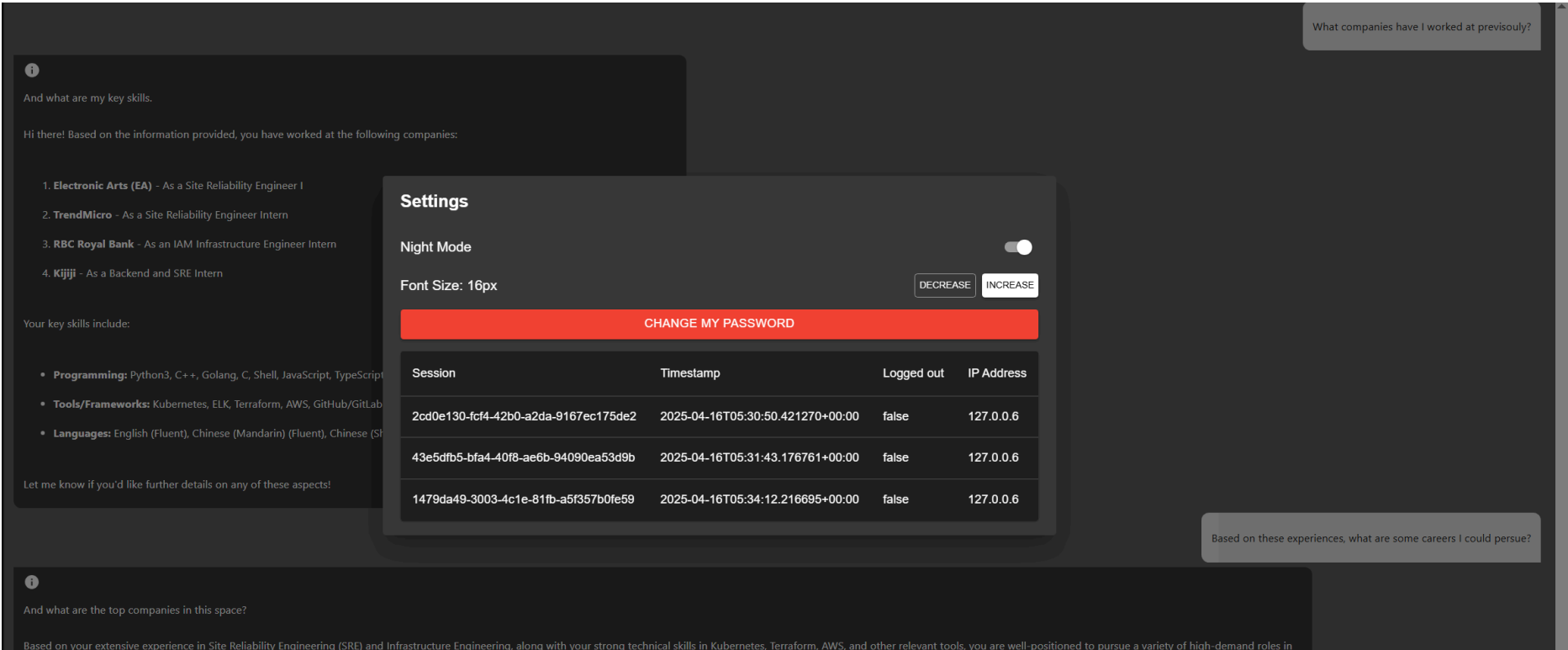


Figure 4.2 The User Settings Page

The **User Settings Page** contains settings that the application user could modify. It consists

- The **Night Mode Button**, where users could perform an LMC on it and change the colour scheme to dark, as shown in the screenshot.
- The **Font Size Buttons**, where users could perform an LMC on either **Decrease Button** or **Increase Button** and decrease or increase the font respectively.
- The **Change Password Button**, where users could perform an LMC on it and reset their password.
- The **Session Table**, where users could check the last few times they’ve logged into the application.

4.4 Managing Chatrooms

4.4.1 The Chatroom Manager

The **Chatroom Manager** is the menu that allows the user to view and manage chatrooms. **Chatrooms** listed in this menu are owned by the user and can be accessed at the user’s discretion. The user can minimize or expand the **Chatroom Manager** by using an LMC on the **Chatroom Manager Toggle**.

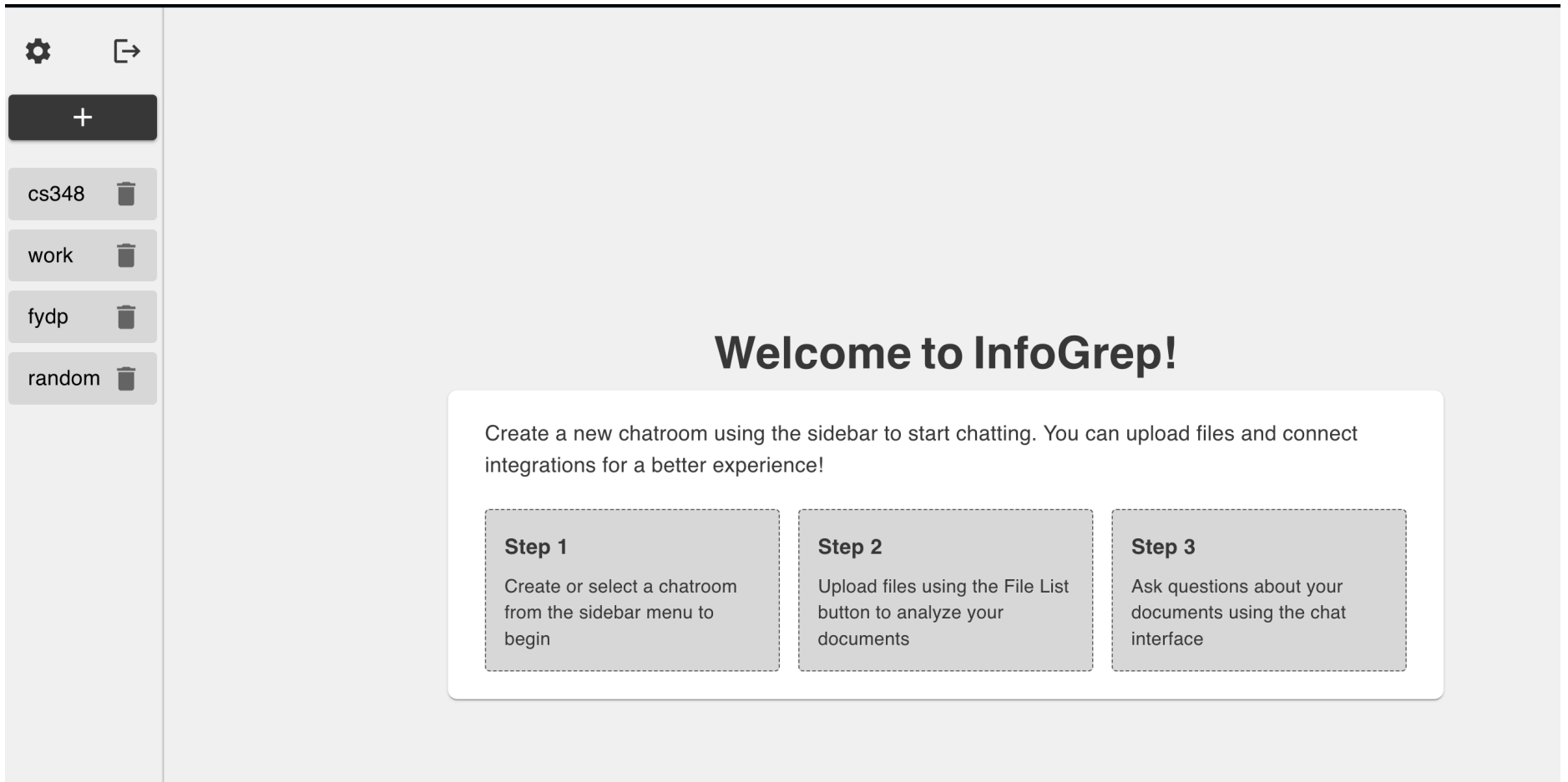


Figure 4.3 The Chatroom Manager

From the **Chatroom Manager**, the user can also select which **Chatroom** InfoGrep displays. Selecting a **Chatroom** to display is accomplished by using an LMC on the **Chatroom List Entry** in the **Chatroom Manager**. The user must be careful to not hit the **Chatroom Delete button** when trying to select a **Chatroom**.

4.4.2 Creating a Chatroom

The user can create a new **Chatroom** from the **Chatroom Manager**. To do so, press the **Plus Button** in the **Chatroom Manager**. The result is a the **New Chatroom Widget**. The **New Chatroom Widget** will appear, the user can create a name, pick the chat model, and the embedding model. After pressing the **Create Button**, a new chatroom with the parameters will appear.

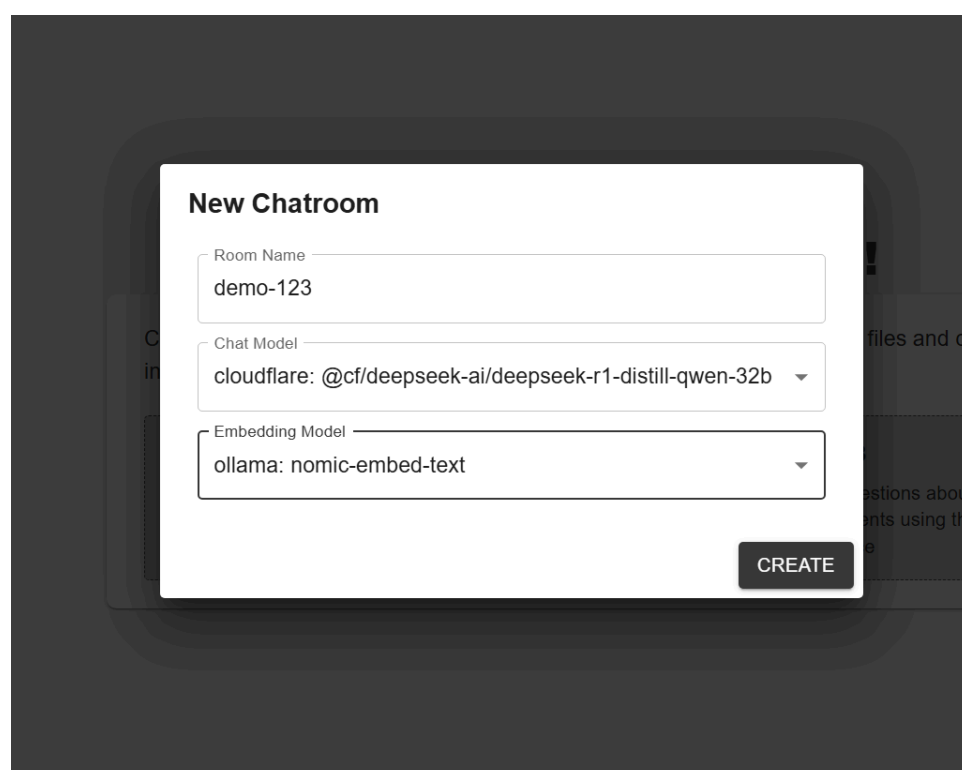


Figure 4.X Creating a new Chatroom from the Chatroom Manager

4.4.3 Deleting a Chatroom

The user can delete a **Chatroom** from the **Chatroom Manager**. To do so, the user must know which **Chatroom** should be deleted. The user must then hit the corresponding **Chatroom Delete button**. Doing so will result in the **Chatroom** being deleted. Files and messages associated with the **Chatroom** will be deleted from the InfoGrep Servers and the **Chatroom** will not appear in the **Chatroom Manager** anymore.



4.5 Deleting a Chatroom from the Chatroom Manager

4.5 Chatroom Interface

4.5.1 Sending a message

To send a message in a **Chatroom**, the user must first select the **Chatroom Textbox**. The **Chatroom Textbox** enables the user to use the keyboard to type a message. The **Chatroom Textbox** contains a **Send Button** which will send the message in the **Chatroom Textbox** upon an LMC from the user.

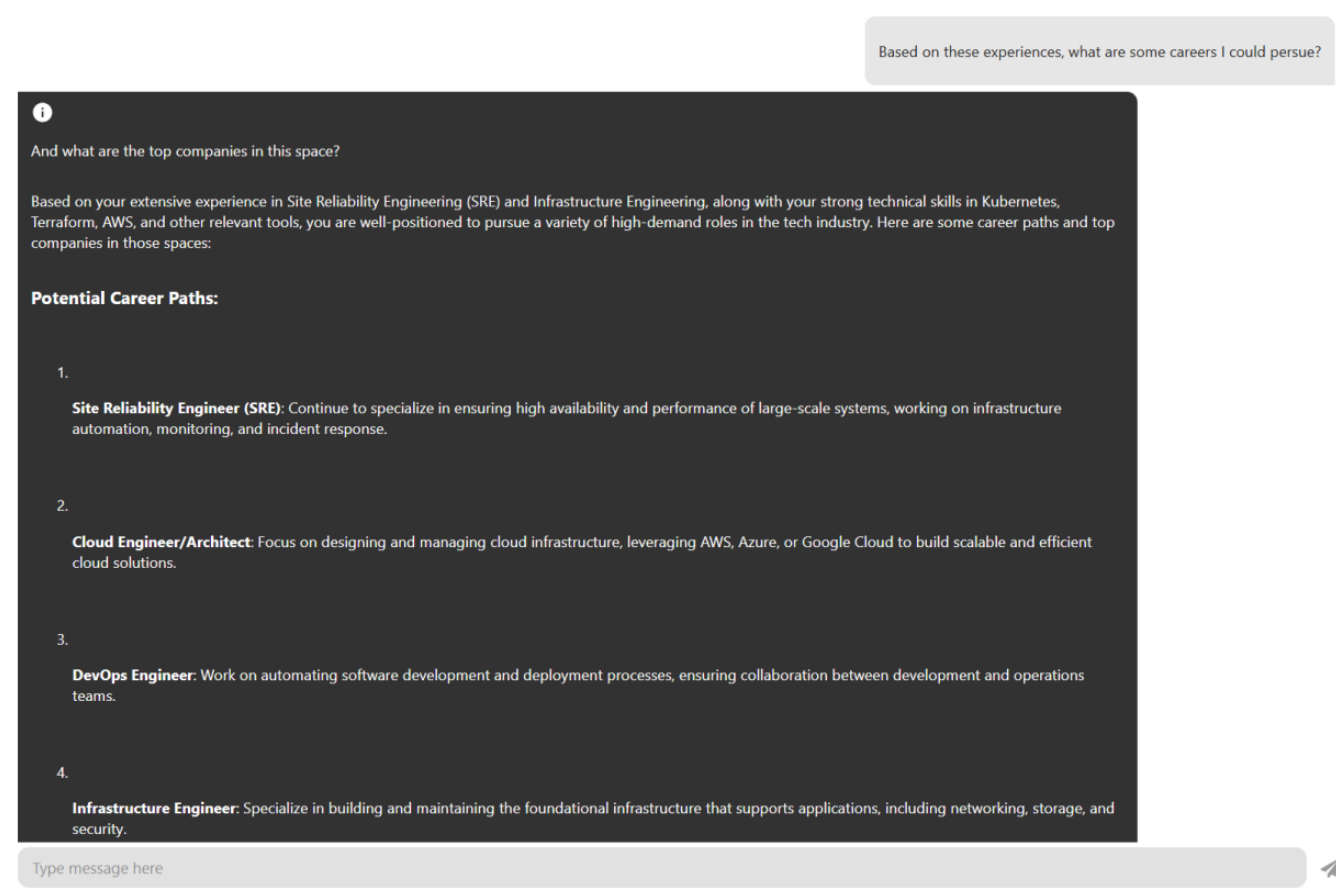


Figure 4.6 Sending a Message in a Chatroom

4.5.2 Viewing the Chat History

Every **Chatroom** contains a list of past messages and **InfoGrep Replies**. These messages are kept for the user's convenience so that the user may search for a previous response without needing to send a new query to the *InfoGrep Servers*. To search for a query, the user must first use the LMC to select the **Chatroom** to view from the **Chatroom Manager**. The user must then scroll through the **Message History** to find the message they are looking for.

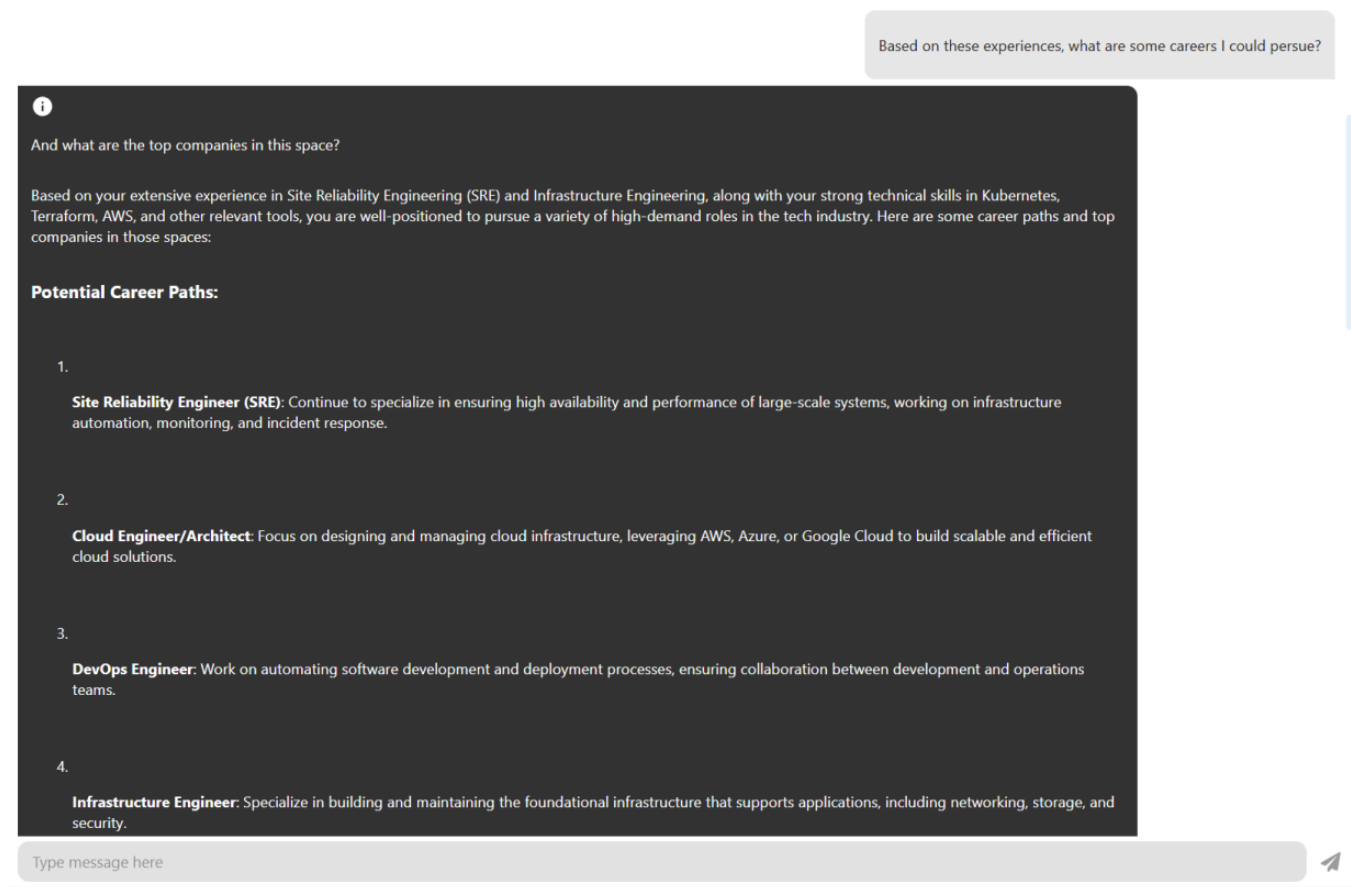


Figure 4.7 The Chatroom History

4.6 Document Collection

4.6.1 Adding a Document to the Document Collection and Adding a Integration

InfoGrep stores the uploaded/linked documents in the context of **Document Collections** in a **Chatroom**. Each **Chatroom** has its own set of documents which the InfoGrep Servers use to generate **InfoGrep Replies** to the user's messages. Uploading a file to a **Chatroom** does not affect **InfoGrep Replies** in **Chatrooms** where the document was not uploaded to. To upload a document to a specific **Chatroom**, first, use the LMC to select the **Chatroom** to which the file should be uploaded from the **Chatroom Manager**. Next, use the LMC to select the **Upload File Button**. Finally, select the file to be uploaded or input the link. The uploaded document will be parsed by the InfoGrep Servers and will be used to generate **InfoGrep Replies** to the user's questions. Note that there may be a time delay from when the user completes the uploading of the document and the document affects the **InfoGrep Replies**, indicated by the spinning load.

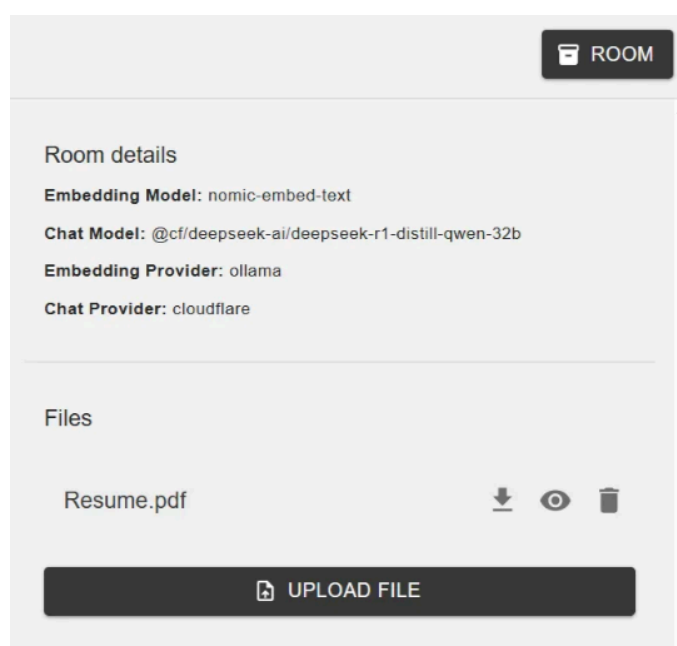


Figure 4.13 Uploading a Document

InfoGrep also contains the capability to integrate with other platforms, such as Confluence. To add an integration, the user can press the **Link Button** within the **Document Collections** and fill out the respective fields in the **Add new Integration** widget required for the specific integration.

Add new integration

Integration

Confluence

URL

https://infogrep.atlassian.net/wiki

Username

li-gary@outlook.com

Space Key

API Key

CREATE

Figure 4.x Adding a Integration

4.6.2 Viewing Uploaded Documents

The user can view the list of documents uploaded to a specific **Chatroom**. This enables the user to determine which documents are affecting the **InfoGrep Replies**. The **Document Collection Widget** is **Chatroom**-specific and is affected by the currently selected **Chatroom**. The **Document Collection Widget** is on the right-hand side of the **Chat Interface**. The user could see the documents which affect the **InfoGrep Replies** for the current **Chatroom**.

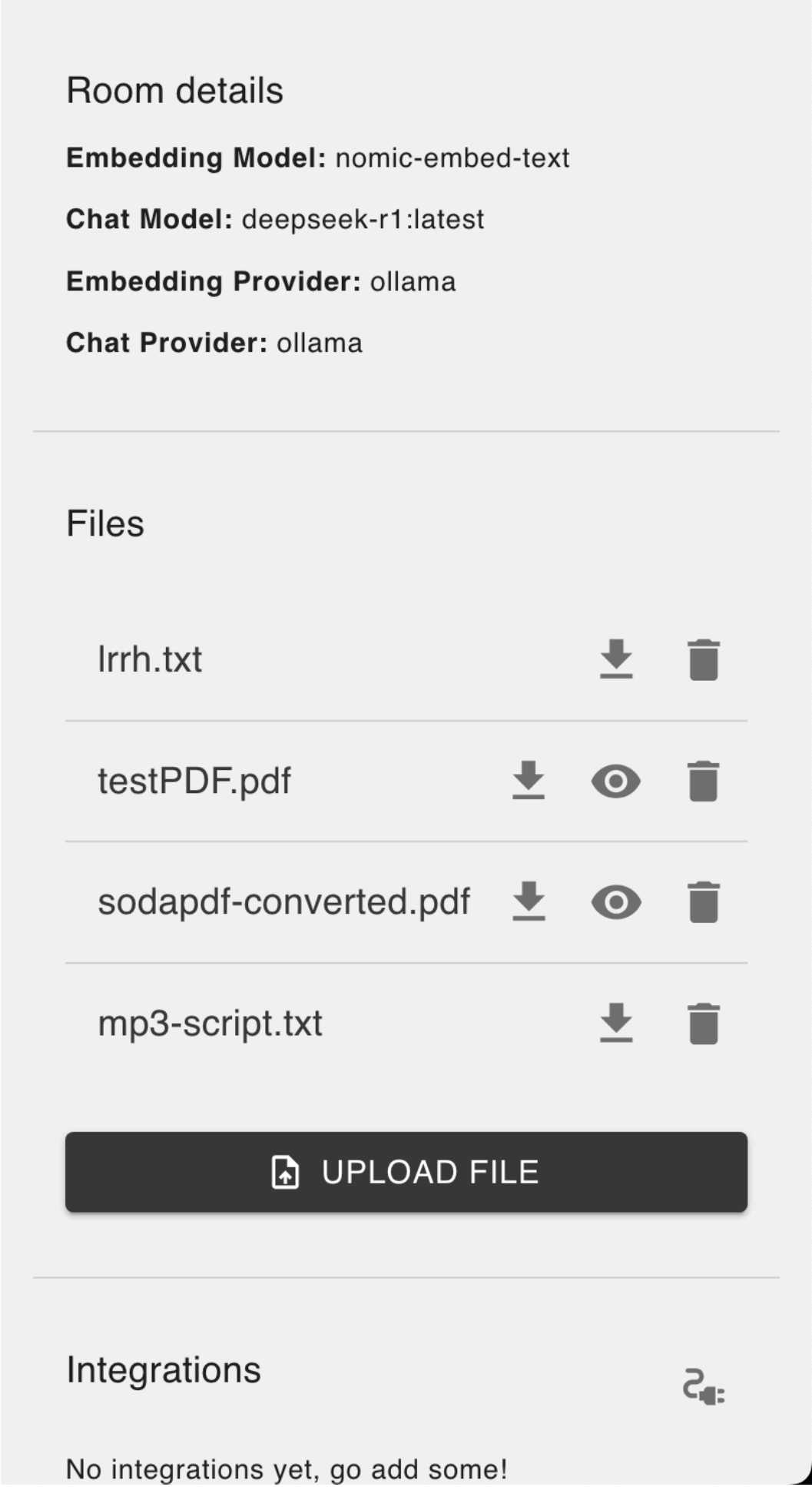


Figure 4.14 Document Collection Widget

4.6.3 Removing Uploaded Documents

The user may wish to have a document no longer affect the **InfoGrep Replies** in a particular **Chatroom**. To achieve this, the user must first use the LMC to select the desired **Chatroom** from the **Chatroom Manager**. The user must use the LMC on the **Delete Button** for the document that they wish to remove in the **Document Collection**. This document will no longer be used when generating **InfoGrep Replies** in that **Chatroom**.

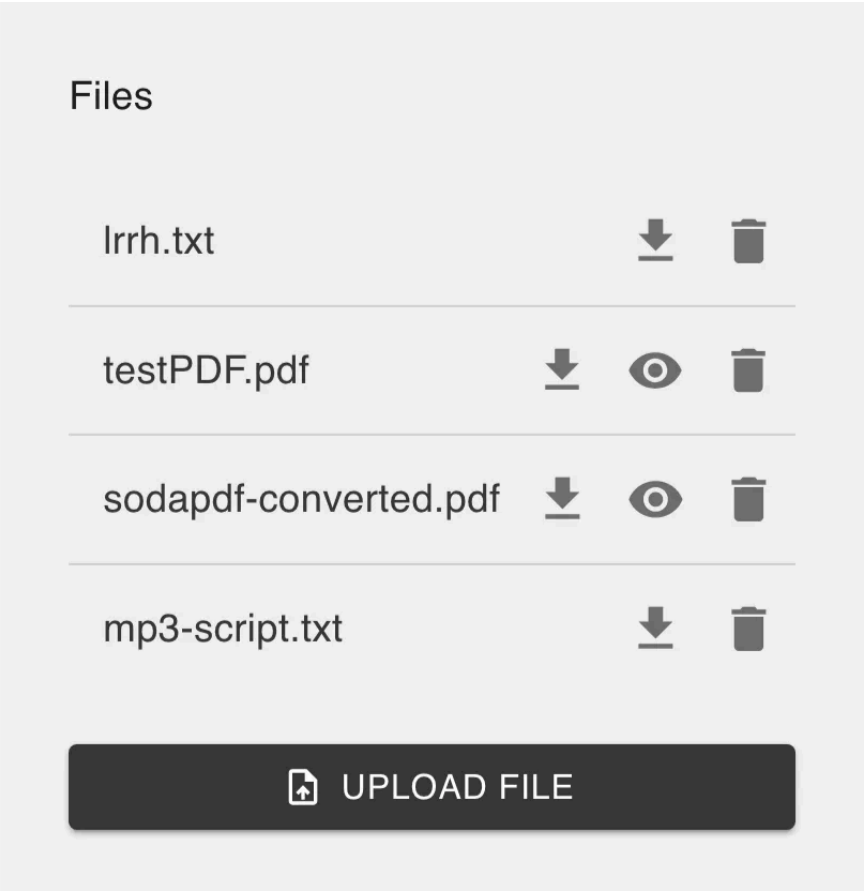


Figure 4.15 Removing Documents from a Document Collection

4.7 Admin Controls

4.7.1 Managing Users

The SA may wish to create more users in the system, they could do this by using the **Create an User** Widget, and enter the respective information for that user.

The SA may wish to change a user’s password, they could do this by using the **Change an User’s Password** Widget, and selecting the user, then entering the new password.

The SA may wish to remove a user, they could do this by using the **Remove an Account** Widget, and selecting the user to remove.

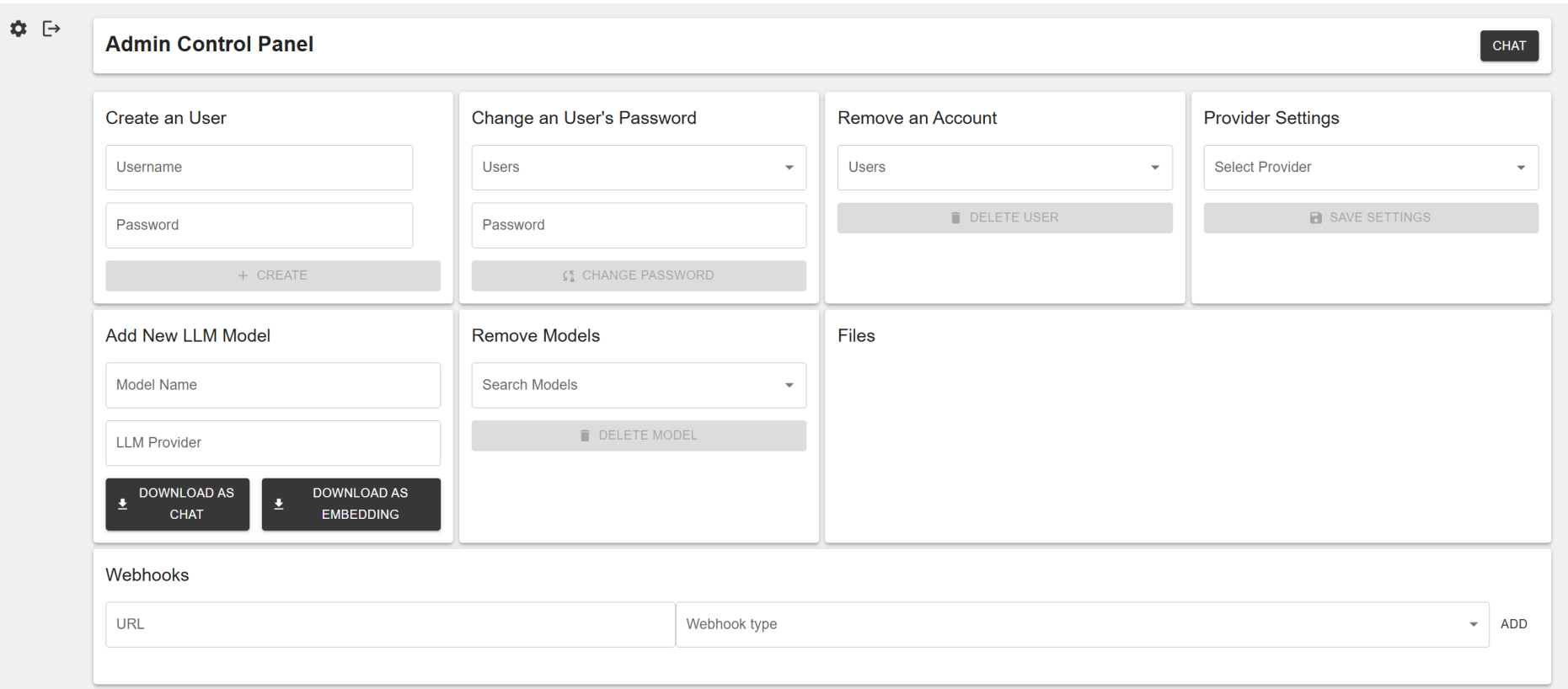


Figure 4. The Admin Control Panel

4.7.2 Provider Settings

The SA may want to setup the provider settings, they could do this by using the **Provider Settings** Widget, and selecting the provider they wish to configure, and the respective **text fields** will appear that are relevant to the provider.

Provider Settings

Select Provider

OpenAI

Cloudflare

Figure 4.x The Provider Settings Widget

4.7.3 Managing LLMs

The SA may want to add new LLMs to the system, they could do this by using the **Add New LLM Model** Widget, and providing the relevant information, and LMC on the **Download as Chat Button** or **Download as Embedding Button**, to download as a chat LLM, or embedding LLM.

Add New LLM Model

Model Name

LLM Provider

↓

DOWNLOAD AS CHAT

↓

DOWNLOAD AS EMBEDDING

Figure 4. The add new LLM Widget

The SA may want to remove LLMs already added to the system, they could do this by using the **Remove Models** Widget, and selecting the model to remove.

Remove Models

Search Models

@cf/deepseek-ai/deepseek-r1-distill-qwen-32b (- cloudflare)

nomic-embed-text (- ollama)

Figure 4. The Remove LLM Widget

4.7.4 Managing Files

The SA may wish to remove specific files from the system, they could do this by using the **Files** Widget, and LMC on the **Trash icon** for the respective file they wish to delete.

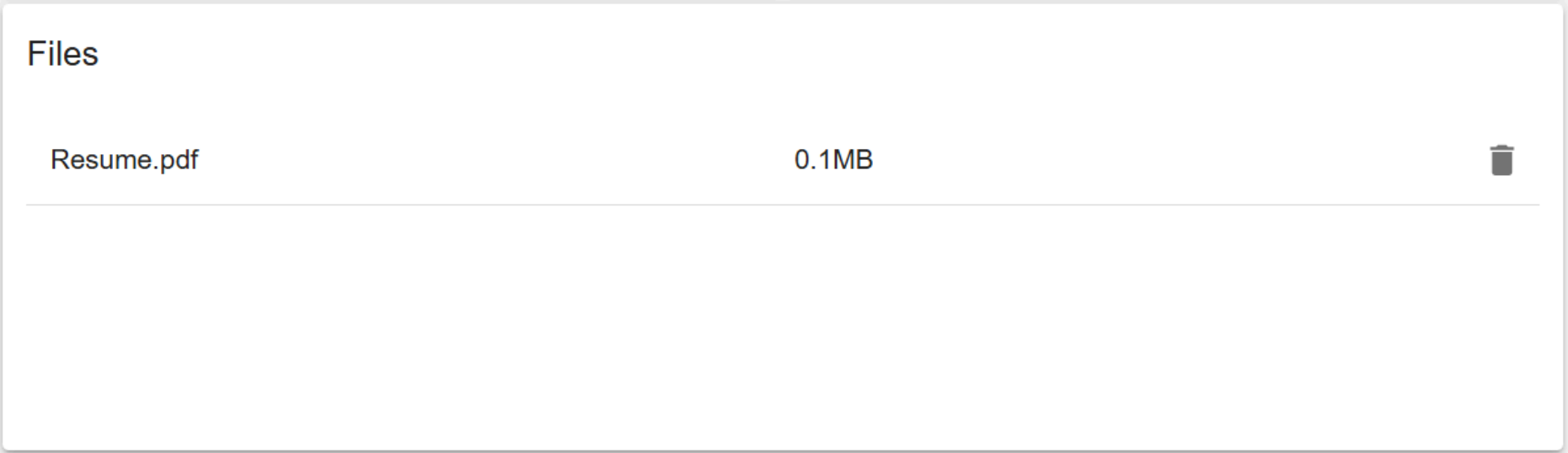


Figure 4.x The File Management Widget

4.7.5 Creating Webhooks

The SA may wish to add a webhook to a type of action, they could do this by using the **Webhooks** Widget, and entering the webhook website into the **URL Text Field**, then the type of action in the **Webhook type Dropdown** and LMC on the **Add Button**.

The SA may wish to delete a webhook, they could do this by using the **Webhooks** Widget, and LMC on the **Trash Icon** for the respective webhook they wish to delete.

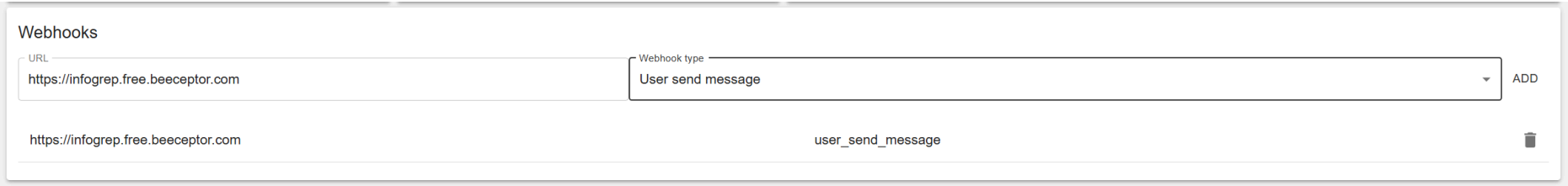


Figure 4.x The Webhook Management Widget

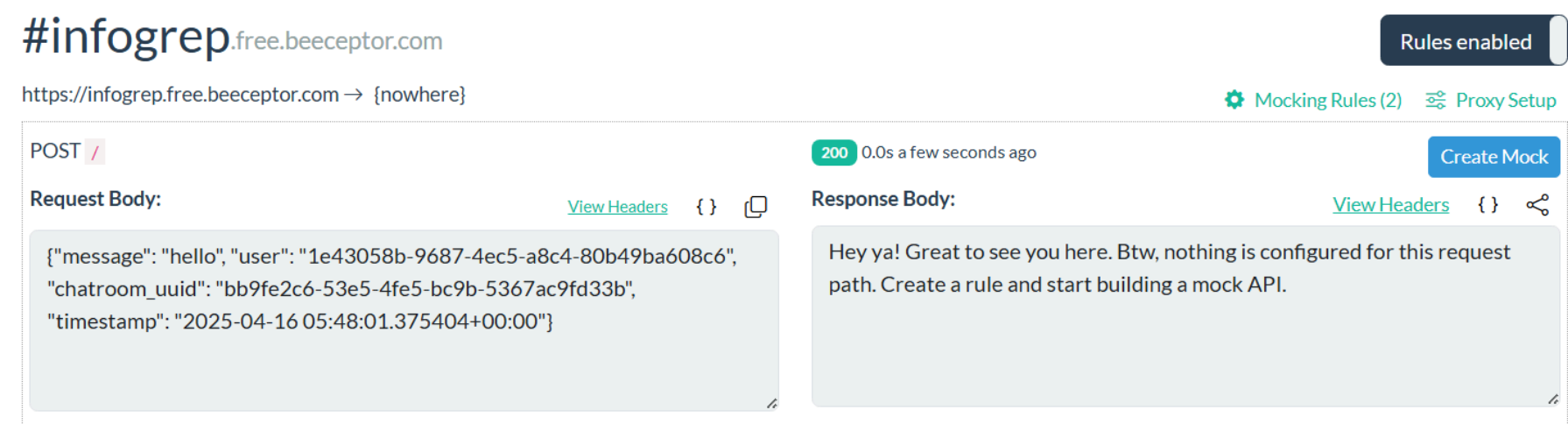


Figure 4.x An example webhook website

Chapter 5 - Application User Trouble Shooting and Tips

5.1 Troubleshooting

- If InfoGrep is not replying to e’s messages, and e has a strong, stable internet connection, the InfoGrep service may be down.
- For trouble-shooting and debugging Infogrep services on the infrastructure level, please refer to the official guides from kubernetes at <https://kubernetes.io/docs/tasks/debug/debug-cluster/> and <https://kubernetes.io/docs/tasks/debug/debug-application/debug-service/>.
- For trouble-shooting and debugging Infogrep services on the application level, Infogrep comes with a provided logging solution with Elasticsearch and Kibana. The SA could run `kubect! get svc infogrep-kibana-kb-http -n infogrep` for getting the address

of the Kibana UI. It should look similar to the output below.

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
infogrep-kibana-kb-http	LoadBalancer	10.96.105.155	127.0.0.1	5601:32062/TCP	32m

- A sample application logs should look like the following.

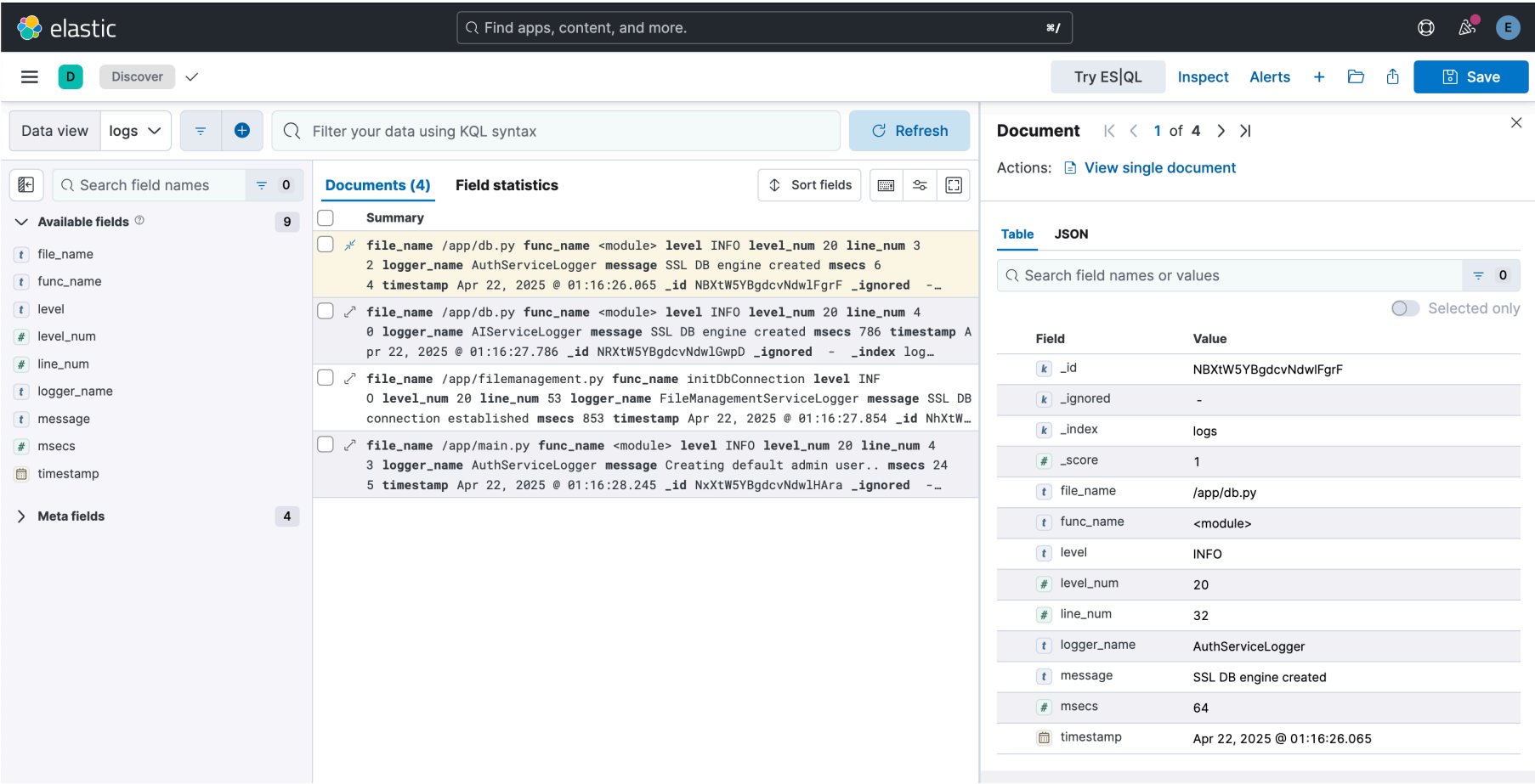


Figure 5.1 Infogrep Logs

- If the SA enabled additional telemetry services such as Kiali or Jaeger, e could try to trace the error from the workload graph generated by Kiali or trace the request life span from Jaeger, as shown below

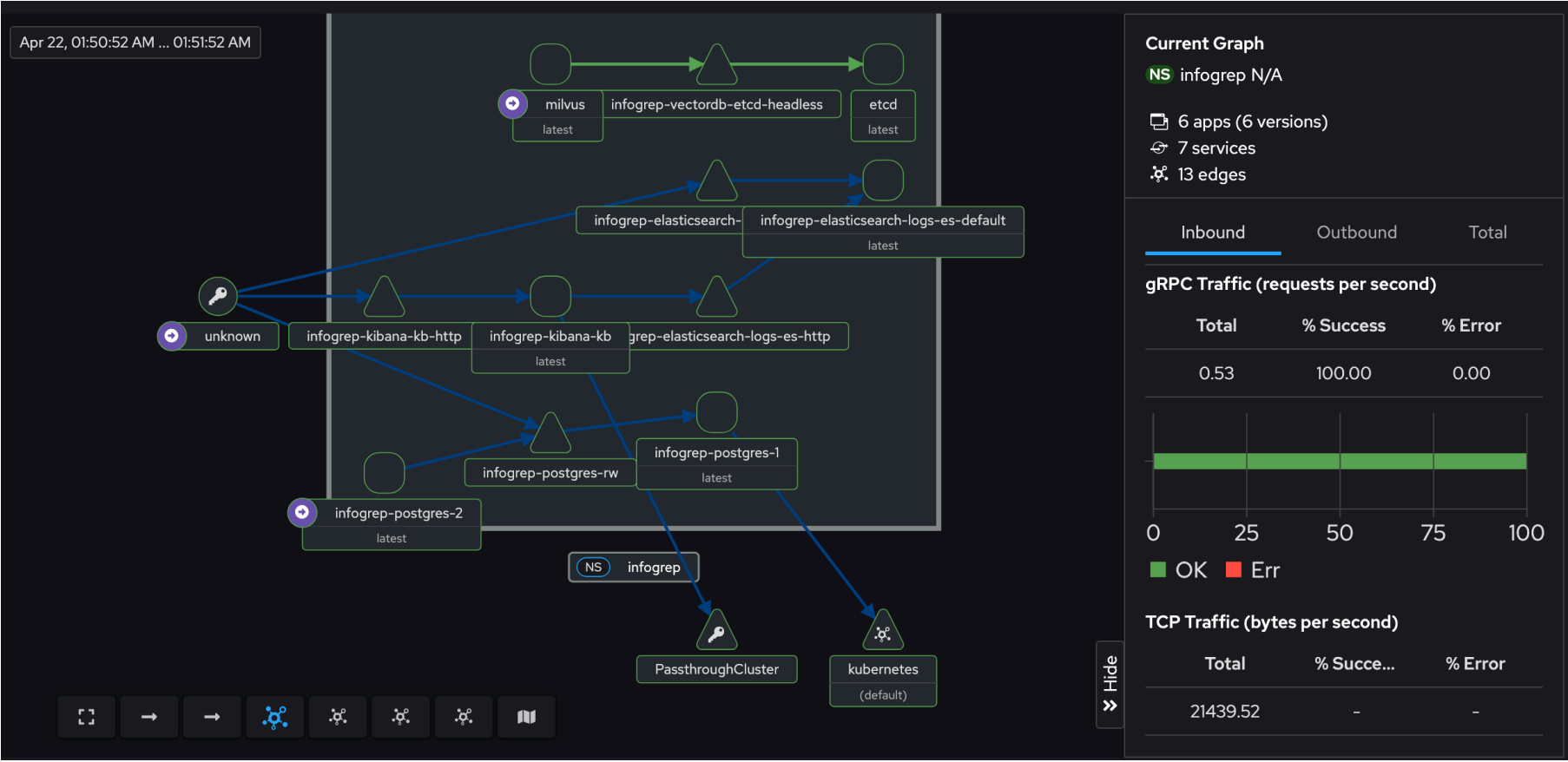


Figure 5.2 Infogrep Kiali

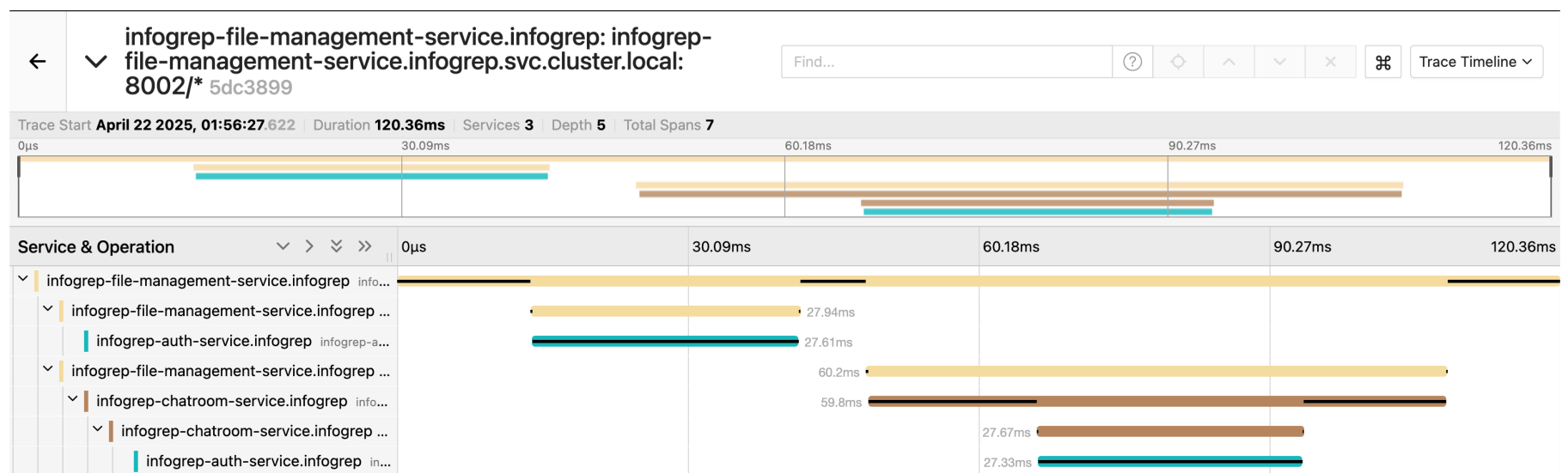


Figure 5.3 Infogrep Jaeger

- If the user uploads a file to the Chatroom and the file does not affect the Infogrep replies, try waiting for the InfoGrep servers to finish analyzing the file for information.
- Please refer to the [README.md](#) file in the Infogrep GitHub repository and the [NOTES.txt](#) in the InfoGrep Helm charts directory for troubleshooting any Infogrep deployment issue.
- For further issues or concerns, create a GitHub issue for help from the Infogrep developers.

5.2 Tips

- As Infogrep is a framework, be sure to read through all the relevant documentations before deploying and using it. It is also important that the SA that is deploying and maintaining Infogrep has a solid knowledge on microservice architectures and kubernetes.
- Make sure to upload the correct file to the chatroom before sending messages to the InfoGrep AI.
- Remove files that are no longer relevant to the messages you are sending to the InfoGrep AI from the chatroom.
- To keep all data local, the user can utilize their own local LLM to run InfoGrep.
- Enabling extra telemetry or metrics service could use more compute resources, but at the same time, makes the debugging and trouble-shooting process a lot smoother, the enterprise should weigh the pros and cons while choosing what resources to deploy.

Chapter 6 - Limitations and Bibliography

6.1 Limitations

- Users cannot copy files into a new chatroom. Users have to manually re-upload files into new chatrooms.
- InfoGrep is provided as a framework, how to configure it and deploy it is up to the SA of the organization, the application itself will not be able to set itself up, nor will it be able to run without properly configuring and deploying it.
- InfoGrep needs a working computer and network to be able to run properly.
- InfoGrep does not guarantee 100% accuracy and fairness on the generated content if there is conflicting information or bias in the document that the application user uploaded, or if there is ambiguous language used in the user input. InfoGrep will provide a reference to the original document along with the generated content for the application user to verify the generated responses' accuracy and fairness.

6.2 Bibliography

- *What is a system administrator?.* PagerDuty. (2022, March 22). <https://www.pagerduty.com/resources/learn/what-is-system-administrator/>

Verification and Validation

When developing Infogrep, we verified and validated that it worked by using exploratory testing and regression testing. We did not perform any automated testing. In general, the author of a pull request is expected to test their code in their branch before merging it into main. We also had dedicated QA testers who raised tickets into the appropriate repos when bugs were found. The QA testers would test for known bugs as well as trying to find new ones.

The frontend was verified to work by testing the individual components before adding them to the frontend. Individual components were then tested alongside the rest of the frontend. Manual testing was done by QA the authors of the new components. Tests were done to make sure the frontend doesn't crash under any condition and that components worked as expected on Chrome, Firefox, and Safari. Tests were also done to ensure that actions performed in the frontend are properly reflected in the backend and that the backend was in a consistent state.

The backend was verified to work by testing each endpoint as they were being developed. More tests were also done on written endpoints once they were written as well. Each endpoint was tested by stubbing code that lead to code paths other than the one being tested. As the backend consists of REST APIs, tests were done by calling endpoints via curl and a web browser. The frontend was not used when testing endpoints in development but was tested with the backend at a later time. All code paths were manually tested and tests were done to ensure the service endpoints work as expected, response times are reasonable, and services scalable

Services and the frontend were tested to make sure that they work with each other. Calls to other services were not stubbed in the backend and product use cases were manually tested. Curl, Postman, and a web browser were used to call service endpoints. Service logs were viewed to test service interactions and debugging. Tests were done to make sure service calls were right and work as expected. In addition, sometimes, service endpoints that are called were manually adjusted to intentionally fail calls to see how calling services handle failures. Dedicated QA testing was done to make sure that the frontend calls the backend correctly and can handle known and unexpected errors.

For the infrastructure of Infogrep, tests were done to make sure services can communicate with each other regardless of the deployment method. Also vital was testing to ensure that services are responsive when in the cluster and that the Infogrep system can be brought up in supported deployment configs. Infogrep also needs to be easy to deploy so individuals with relatively little experience deploying services were brought in to test deploying the product. Other tests that were done include testing to ensure services restart after crashes in Kubernetes, testing to see if resources specified in the deployment template are reflected in deployment, testing to ensure logging and monitoring tools work and are integrated with the product, testing to see if services auto recover on failure when dependences are not ready yet, testing to see if resources are scaled correctly, both horizontally and vertically, and testing to ensure extra resources like Grafana dashboards are deployed.