SavvyK8s

Neha Poonia - 015252900
Department of Software Engineering, San Jose State University´San Jose, CA
neha.poonia@sjsu.edu

Sarvesh Upadhye - 015220855
Department of Software Engineering, San Jose State University´San Jose, CA
sarvesh.upadhye@sjsu.edu

Bharath Jagini - 015260232
Department of Software Engineering, San Jose State University´San Jose, CA
bharath.jagini@sjsu.edu

Anuj Sharma - 015340130
Department of Software Engineering, San Jose State University´San Jose, CA
anuj.sharma@sjsu.edu

ABSTRACT

Containerization of applications is now a new normal way of deploying applications efficiently. Kubernetes is presently one of the big market competitors with that of RedHat's OpenShift and IBM. Kubernetes being Open Source is very much popular among the community. This orchestration tool also has built in CLI support to monitor metrics. SavvyK8s is an application to visualize and monitor the K8s cluster resources through a graphical user interface. The application will have real time usage charts, and will notify users via email or browser notifications on reaching thresholds as set by the users.

1 INTRODUCTION

Kubernetes or K8s is an open-source container-orchestration system for automating computer application deployment, scaling, and management[1]. Starting with a collection of Docker containers, Kubernetes can control resource allocation and traffic management for cloud applications and microservices. As such, it simplifies many aspects of running a service-oriented application infrastructure . The default way to handle and monitor different K8s is through command line interface. CLI makes it difficult to monitor multiple aspects of the cluster at once, in a real-time way. Monitoring [2]is required to anticipate problems and have visibility over potential bottlenecks. Our objective is to create a web UI based monitoring tool where users will be able to monitor[3] different resources (cluster, pod, application etc.) through a web UI.

2 RELATED WORKS

2.1 Monitoring Kubernetes Cluster with Prometheus

Prometheus is an open source platform for containers and microservices. Prometheus aims to collect numerical data based on time series data . It sends an HTTP request, its server works on the principle of scraping . The response to this scrape request is stored and parsed in storage along with the metrics for the scrape itself. 15 days is the data retention time by default while the lowest retention time is 2 hours . The storage is a custom database on the Prometheus server and can handle a massive influx of data. It's possible to monitor thousands of machines simultaneously with a single server.

2.2 Grafana

Grafana is a multi platform visualization software that helps us visualize and query our data in the most powerful way . It has a number of options for client side graphs with many plugins available to visualize metrics and logs . Its different kinds of dashboard helps companies fully understand the analytics of their users and events related to their infrastructure .

3 ARCHITECTURE

This section deals with the components of the architecture .[4] The main architecture flow can be observed from Figure 1 .

3.1 Database-MongoDB

MongoDB is an object oriented document type database which stores data in JSON like documents instead of storing it in columns and rows . It gives developers a fast and powerful way to store and retrieve data . As MongoDB has schema less operations it is useful when the data structures need to evolve with time . The main features of MongoDB are :

- Ad-hoc queries
- Indexing
- Aggregation
- Load balancing
- Transactions

We used MongoDB for our application as it is a non relational database and also schema free. We use JSON responses throughout the application and storing it as documents in collections would be an ideal choice. Moreover, there is ample amount of scope to extend functionalities other than memory and cpu metrics with different types of graphs. NoSQL is a perfect fit for us due to these challenges we may encounter if we opted for a Relational database. We used atlas MongoDB cloud for storing data in MongoDB.

3.2 Server implementation

We used Node Js backend for producing API's. Whenever a user logs in from web application React.js in our case we call Node.Js backend API and check authentication. If authentication is successful we generate JWT token and send token along with response. For each request after login/signup, token needs to be passed for authorization. Mongoose was used as Object Data Modeling to connect to MongoDB from Node Js. For storing images,Amazon s3 bucket was used to store it in cloud.

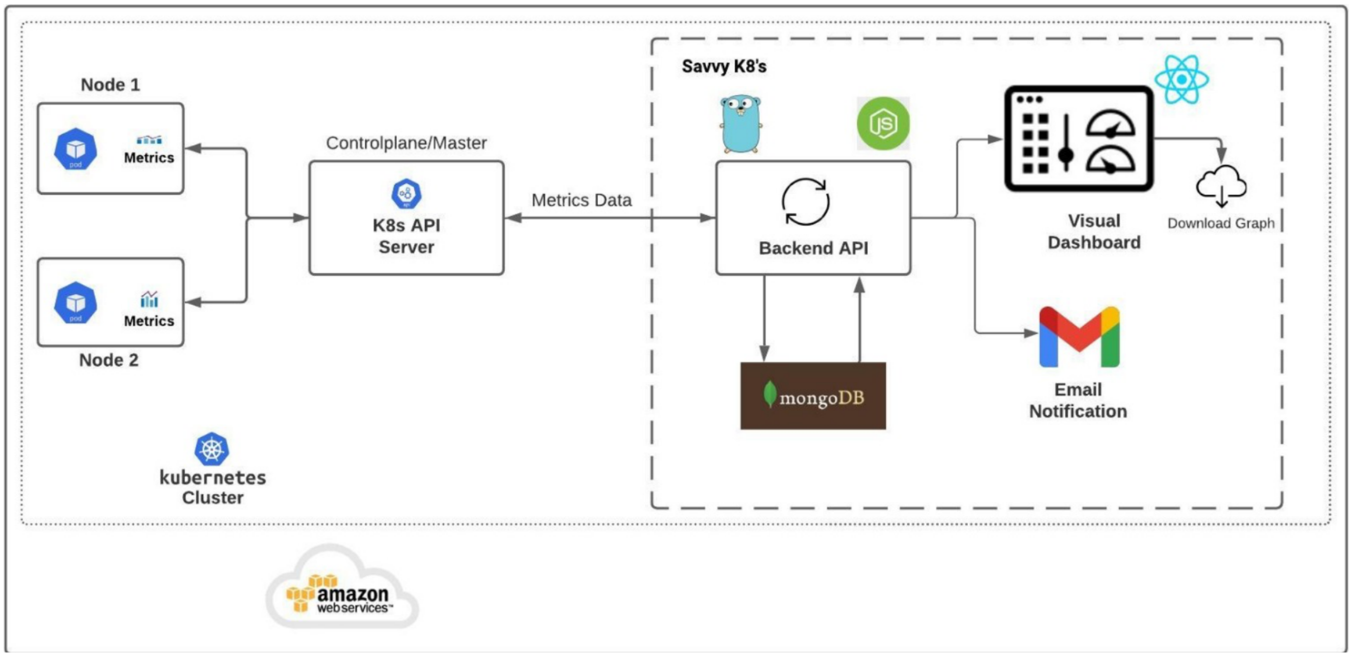


Figure 1: Architecture of our proposed system

3.3 Deployment

The Application is divided into 2 major parts that are frontend and backend. Both the modules are deployed on AWS EC2 machines. The Kubernetes Cluster [3] (i.e. 1 Controlplane and 2 nodes) were deployed on "t2-medium" type of machine as kubernetes requires at least 2 cpus and 4Gb of memory. The Frontend, that equipped React application, was deployed on a t2-micro instance, as it needed very little computing power.

4 FUTURE ENHANCEMENT

Following are the enhancements that we look forward to implement in the near future .

- Add support for Slack and discord
- Add functionality for importing namespaces from clusters and implementing Role Based Authentication (RBAC) for the application
- Add Different types of graphs and data visualisation options
- Add support to analyze and record metrics caused, from UI .

5 CONCLUSION

With our GUI application we were able to solve the problem of monitoring K8s clusters in a more efficient way as compared to command line interface . A trigger email for memory and CPU threshold can help administrators check and solve the errors instantaneously . The application would be helpful for DevOps engineers , System administrators , Project managers and also for the application testing team as a whole .

6 CONTRIBUTIONS

Sarvesh Upadhye : Worked on kubernetes cluster setup and designing backend part in golang to gather metrics and store to database. Developed Sending automated mail feature using Gmail API. Assisted in deploying the backend part of the project on AWS.

Bharath Jagini : Worked on UI development with React and Node JS for backend. Assisted on deploying web applications cloud (AWS). Added feature to download realtime graph metrics for the cluster.

Anuj Sharma : Worked on designing and developing UI part of web application in react and also assisted in developing backend in Node JS. Worked on testing part of the application for bugs and non-functional features.

Neha Poonia : Designed the User flow and User Interface for the application and also assisted with developing the frontend of the web application in React. Worked on Testing part of the application to find and fix bugs and features.

7 SCREENSHOTS OF THE APPLICATION

This section contains the images of the product in action . Following is an index to follow each of the image :

- Figure 2 consists of Landing Page
- Figure 3 consists of Login Page
- Figure 4 consists of Signup Page
- Figure 5 consists of Dashboard Page
- Figure 6 consists of Node Metrics(Area Chart)

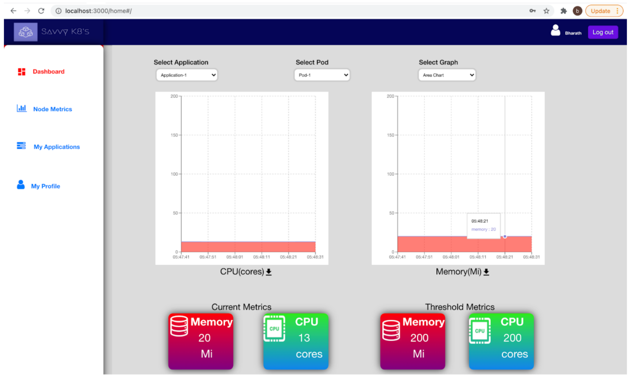


Figure 5: Dashboard Page

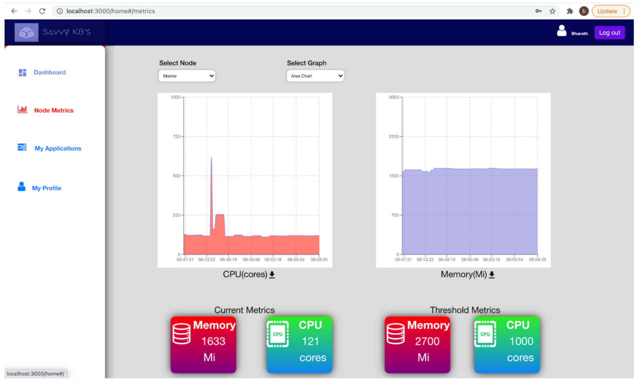


Figure 6: Node Metrics(Area Chart)

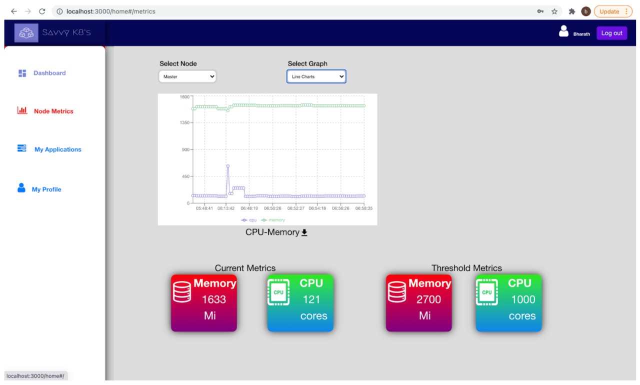


Figure 7: Node Metrics (Line Chart)

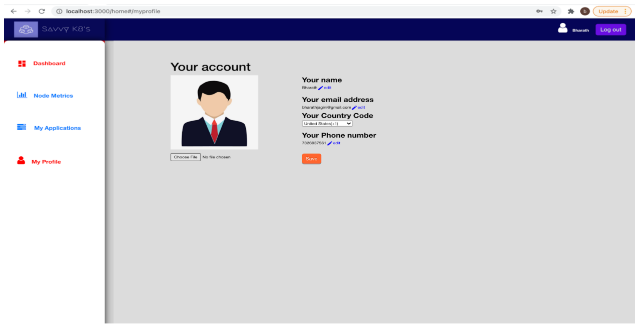


Figure 8: My Profile

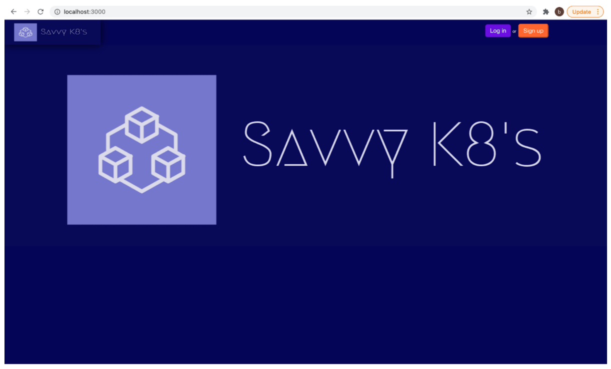


Figure 2: Landing Page

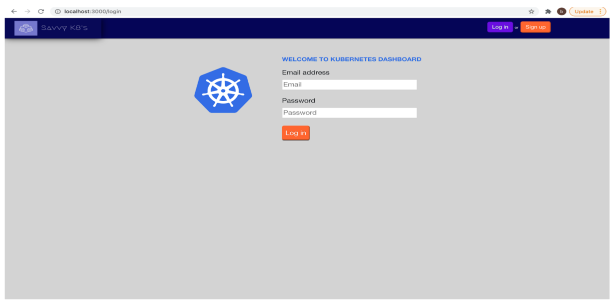
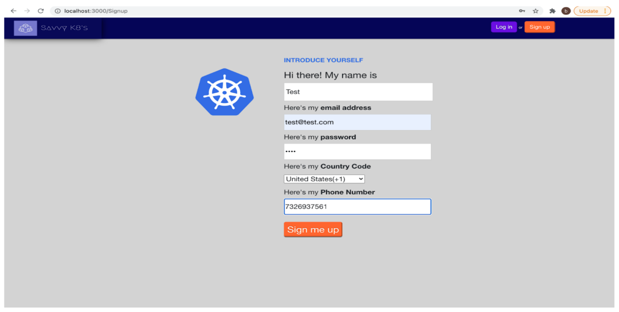


Figure 3: Login Page



Hi there! My name is
Test
Here's my **email address**
test@test.com
Here's my **password**
....
Here's my **Country Code**
United States(+1)
Here's my **Phone Number**
7326937561

Figure 4: Signup Page

- Figure 7 consists of Node Metrics(Line Chart)
- Figure 8 consists of the My Profile Section
- Figure 9 consists of the Downloaded Graphs

REFERENCES

[1] Kelsey Hightower, Brendan Burns, and Joe Beda. 2017. *Kubernetes: Up and Running Dive into the Future of Infrastructure* (1st ed.). O'Reilly Media, Inc.

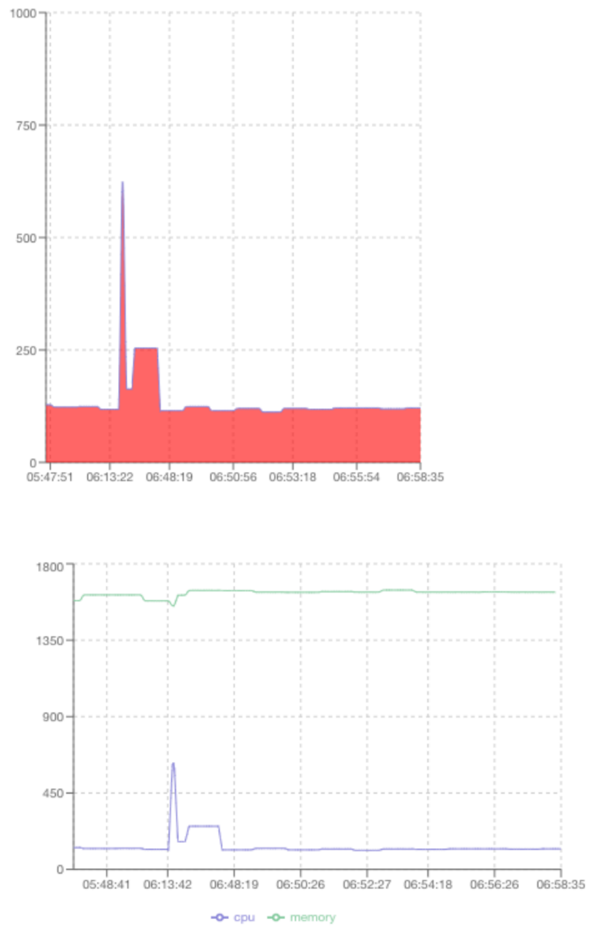


Figure 9: Downloaded Graphs

[2] John Matson Jean-Mathieu Saponaro. 2017. *Monitoring Kubernetes performance metrics* (1st ed.). Datadog 2021.

[3] John Matson Jean-Mathieu Saponaro. 2020. *Collecting metrics with built-in Kubernetes monitoring tools* (1st ed.). Datadog 2021.

[4] Cong Liu. 2019. *Guide to Kubernetes Metrics* (1st ed.). medium.