Internship at Samsung Research Institute, Bangalore

"AIOps - AI/ML enabled operations"

Submitted by:

Dhruv Vohra **PES1201700281**

Under the guidance of:

Adarsha Ananda

Architect, Server Platform, Voice Intelligence

Samsung Research Institute, Bangalore

January 2021 - June 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

100 ft. Ring Road, Bengaluru - 560 085, Karnataka, India

ACKNOWLEDGEMENT

I would like to express my gratitude to my guide Mr. Adarsha Ananda, Architect in Server Platform under the voice intelligence group for his continuous guidance, assistance and encouragement throughout the development of this project.

I am grateful to the internship coordinator Prof. Preet Kanwal, Dept. of Computer Science and Engineering, PES University for organizing, managing and helping out with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department.

I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M.R. Doreswamy, Chancellor - PES University, Prof. Jawahar Doreswamy, Pro Chancellor - PES University, Dr. Suryaprasad J, Vice Chancellor - PES University for providing me various opportunities and enlightenment every step of the way.

Finally, this Internship could not have been completed without the continual support and encouragement I have received from my parents and my friends.

ABSTRACT

This report outlines my work during the internship at Samsung Research Institute, Bangalore (SRIB) over a course of 6 months from January 2021 to June 2021. Samsung Research is the advanced research and development (R&D) hub of Samsung's Consumer Electronics (CE) Division and IT & Mobile Communications (IM) Division. Samsung Research Institute, Bangalore offers internship opportunities to students from prestigious colleges and universities in various departments like Voice Intelligence, 5G, On-device AI, etc. I consider myself fortunate to be able to clear the bar and get an opportunity to work at Samsung Research Institute.

The purpose of this internship is to understand the practical aspects of the theoretical subjects studied at the university, to understand the working of a tech giant in the industry, to gain exposure in the corporate environment and fulfill the credits requirement in the 8th semester of B.Tech in Computer Science and Engineering.

During my Internship I was assigned to the Server Platform team under the Voice Intelligence Group. The Voice Intelligence team from Samsung Research Institute, Bangalore, has contributed significantly in the development of SVoice and Bixby 2.0 (The Voice Intelligence agents of the Samsung ecosystem).

My role as an intern was to develop an end-to-end service from scratch named "AIOps - AI/ML enabled operations". The work included applying the knowledge of data science, machine learning based algorithms and cloud computing technologies. The detailed information about the work, the tools and technologies used and implementation details are mentioned in depth in the later sections of this report. I gained a lot of knowledge on the same from my team and multiple online resources during the development process.

In conclusion, this was an opportunity which greatly helped me to enhance my knowledge and competencies in the AI, ML and cloud computing domain.

TABLE OF CONTENT

LIST OF TABLES AND FIGURES

1. About the company

The Samsung group is a South Korean multinational conglomerate headquartered in Seoul, South Korea. Founded in 1938, Samsung was originally started as a trading company but over the next three decades, the group diversified into numerous affiliated businesses under the Samsung brand. The subsidiaries include Samsung Electronics, Samsung Engineering, Samsung C&T Corporation, Samsung Heavy Industries, etc.

Samsung Research is the advanced research and development (R&D) hub of Samsung's Consumer Electronics (CE) Division and IT & Mobile Communications (IM) Division. Samsung Research leads the development of the future technologies with more than 10,000 researchers and developers working in overseas R&D centers. Core research themes at Samsung Research include artificial intelligence (AI), data intelligence, next-generation communications, robot, Tizen, life care & new experiences, next-generation media, and security. In particular, it is expanding its research scope to new promising fields to realize a new lifestyle based on AI technologies.

The hub is currently working in collaboration with 14 overseas R&D centers in 12 countries worldwide and 7 global AI centers to secure innovative technologies and enhance its global R&D capacity.

Samsung R&D Institute India-Bangalore (SRI-B) is the largest R&D Center outside of South Korea and a key innovation hub in the Samsung group. The specific purpose of SRI-B in the Samsung family is twofold: to create USPs for global flagship devices by creating significant advancements in Modem, Multimedia, AI, Internet of Things, and to make for India by catering to the specific needs of Indian consumers.

2. My role as an intern

I was shortlisted as an R&D intern at Samsung Research Institute, Bangalore (SRIB) after clearing the online coding test followed by 3 rounds of interviews with the friendly and supportive engineers and management staff at SRIB.

After joining the company, I was assigned to the **Server Platform team** under the **Voice Intelligence group**. The group has made significant contributions to the SVoice and Bixby 2.0 which are the voice intelligence agents in the Samsung ecosystem.

Bixby 2.0 like any other AI powered voice-based agent contains multiple components like the **Automatic Speech Recognition (ASR)** component, the **Text to Speech (TTS)** component, the **Personal Data Sync Service (PDSS)** component, etc. Each of these components have separate teams responsible for its development. The work of the server platform team is to integrate these components together and deploy the application in production. The nature of the work in the server platform requires the employee to have a good understanding of the different components that make up the voice intelligent agent.

Apart from the integration of the components and deployment of the application, the server platform team also identifies potential bottlenecks in the services, anomalies and outliers in the cloud instances, and performs log analysis when errors are encountered in the application. However, these are done with the help of **third-party applications like Sumo Logic and Datadog** which are primarily analytics tools for cloud monitoring and give real time insights.

As part of the projects for 2021, Samsung Research Institute, Bangalore, wanted to develop an in-house **end-to-end Service for Bixby 2.0** to perform all the activities for which third-party applications were being used. The project is initiated as a POC (Proof of Concept) to implement and test the feasibility and practical potential of such a service.

The project is titled **"AIOps - AI/ML enabled Operations"** that would allow the developers to analyze the different components and get insights into the same once the application was deployed. I was given the responsibility of researching the various functionalities required in AIOps and implementing the same from scratch along with the development of the E2E service and deployment on a cloud instance.

3. Scope of the project

As explained in the previous section, the server platform team integrates the various components (ASR, TTS, PDSS) that comprise the core of the Bixby engine and deploy the same into the production environment. On the other hand, the server platform team also monitors various metrics of the application, identifies bottlenecks, monitors health status and captures outliers and anomalies using third-party applications like Sumo Logic and Datadog.

The project **"AIOps - AI/ML enabled Operations"** is targeted at implementing few of these functionalities in an efficient way that can reduce dependencies and restrictions due to third party applications and can be designed in a way that is suitable to the employees and developers of Samsung Research Institute, Bangalore while enabling them to better analyze the application and its performance in a convenient way.

4. Project abstract

AIOps - AI/ML enabled Operations is developed to assist developers in Voice Intelligence group to track and analyze the performance of their application. The features include the following:

I. **Metric Forecasting**: When a service or an application is deployed on cloud and goes live for end users a lot of metrics are generated based on the performance and the usage of the service/application. These metrics can be categorized into two main types:

   A. **Application metrics:** These are metrics collected over a period of time that indicate the **performance of the application** itself. Example of application metrics include - number of concurrent requests hitting the service end-point per second, number of new user registrations per day, average time taken to execute a database query.

   B. **System metrics:** These are metrics collected over a period of time that indicate the **performance of the system (virtual machine, cloud instance, docker, etc.)**. Example of system metrics include - average time taken to read/write from

secondary storage, network performance, available memory, consumed memory, available hard disk space, etc.

II. **Anomaly Detection**: The metrics referred to in the previous point usually have a minimum and maximum threshold within which the metric and the application is considered to be working as expected. However, at times a metric or a group of metrics can potentially exceed the min/max threshold which ultimately leads to **degradation in the performance** of the application. In some cases, it might also lead to the **application crashing**. Hence such an anomalous behavior of a metric needs to be identified and reported to the responsible team that can take a quick action to keep the operations smooth and avoid any hindrance at the end users point.

III. **Outlier Detection**: Usually an application is deployed on multiple instances to balance the load to provide a better user experience as the latency to receive a response reduces. Therefore, in a group of instances, all contain the same application source code and serve the same set of end points and hence are expected to have similar system metrics. **Any particular instance that performs differently from the group is marked as an outlier.** It leads to performance degradation for a set of users which the particular instance was serving and it needs to be avoided. Such an outlier is identified in the initial stage of deviation in system metrics so that the responsible team can take a corrective action.

5. Advantages of the project

The advantages of AIOps are as follows:

I. The main advantage of AIOps is to provide developers an application that is **tailored to their requirements** and helps them analyze and get insights about a particular component, a particular metric or a particular instance in production and resolve any issues to provide a seamless and better experience to the end-users from a performance point of view.

II. Another advantage of AIOps is that it **removes the dependencies on third-party** analytics and real time insights applications that would not adjust to a particular developer's feedback.

III. It also helps the company from an **expense** point of view as the third-party applications like SumoLogic and Datadog require a usage-based fee for commercial use. AIOps implement the same features and require much lesser cost for development and maintenance.

6. Technologies used

The Technologies used for the development of AIOps is listed below:

I. Data Science and Machine Learning:

A. Tensorflow
B. Keras
C. Python 3.0
D. Data science libraries in python - Pandas, Numpy, Matplotlib, Sklearn, etc.

II. Web services (Backend):

A. Flask and Flask-RESTful
B. Python 3.0

III. Database:

A. MongoDB
B. Influx DB

IV. User Interface (Frontend):

A. HTML
B. CSS
C. JavaScript
D. Vue.JS

V. Unix utilities:

A. Mutt - for sending emails
B. Cron - for scheduling time-based processes
C. Subprocess - calling bash scripts from python source code

7. High level design and architecture

AIOps is an end-to-end service, end-to-end implies that the user interface (frontend), the API logic and serving requests (backend), the database designs, the automated email alerts and the ML based algorithms required in the working of the said features had to be implemented from scratch.

The **major component** that I worked on was **development of the service** itself (Flask RESTful APIs and the database operations) along with the **Metric Forecasting functionality**.

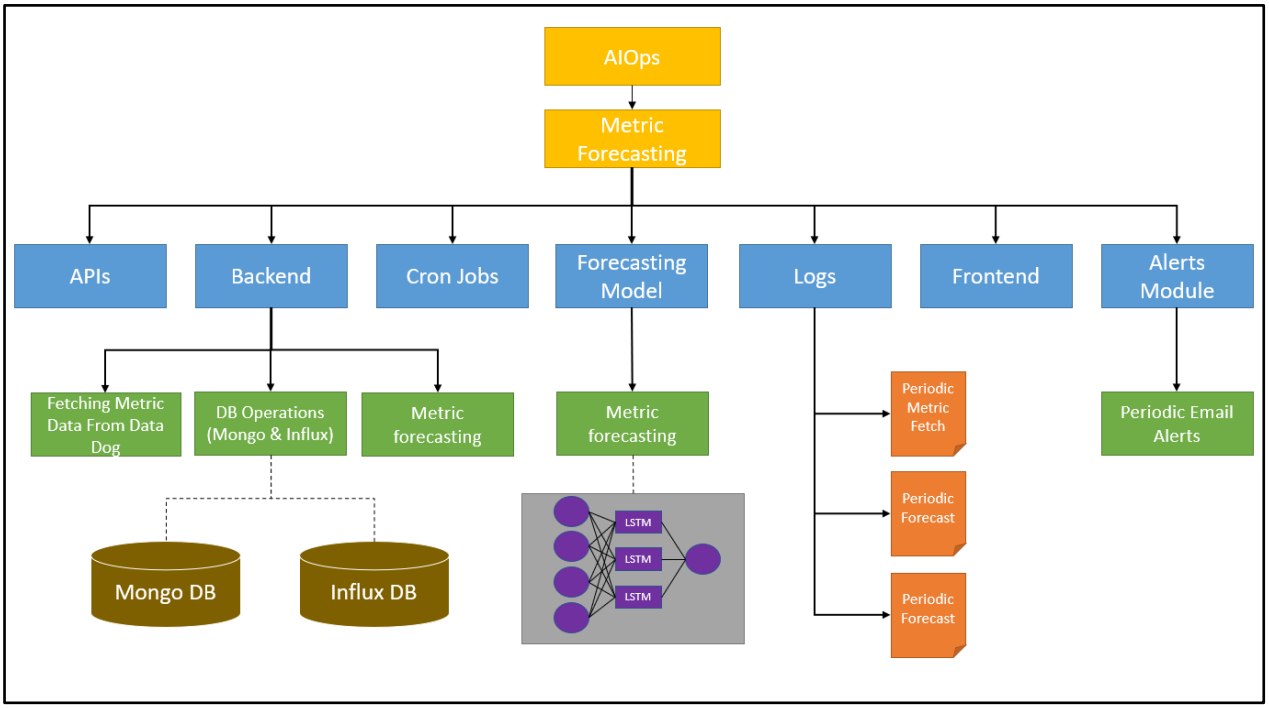

Fig 7.1 - High Level Design (Metric Forecasting)

In Fig 7.1 it can be observed that at a high level the metric forecasting component is divided into 7 sub-components.

The sub-components are responsible for the following:

I. **APIs**: The set of APIs end-points and the associated logic to receive requests from users and serve them accordingly.

II. **Backend**: Further categorized into the following:

   A. **Fetching Metric Data from Datadog**: Although the aim of AIOps was to reduce third-party dependencies, it was implied at their features. The metric data

collection using Datadog agents in any Bixby module had proved to be very consistent and easy to work with in the past. Hence, the readily available Datadog APIs were used to query Datadog with certain arguments and fetch the metric data between the given start and end time.

B. **DB Operations**: It contains the set of processing, reading and writing functions that interact with the underlying MongoDB and Influx DB. MongoDB primarily stores the user details and Influx DB stores the metric values.

C. **Metric Forecasting**: This contains the implementation to forecast a particular metric, includes the calls to data preprocessing before feeding it to the model, the transformation of readable time to epoch time and vice versa and calls to database operations.

III. **Cron Jobs**: Cron is a Unix utility that is used to schedule a process at a particular time. Cron Jobs allow running python programs periodically which is ideal for this project as metric forecasting involves fetching new past data from Datadog, creating new predictions and sending email alerts to registered users at frequent intervals.

IV. **Forecasting Model**: The model which is used for time series prediction. It takes in the past metric data to create future predictions. During the research phase, a number of different models were used to compare performance and efficiency (explained in depth in the next section).

V. **Logs**: It is a good practice to collect logs when an application goes live as it enables the developers to analyze the logs in case an error is encountered or during optimization or maintenance activities.

VI. **Frontend**: The user interface is what the end user interacts with. Frontend needs to be tailored to provide the best user experience and make it easy for the users to get the required information or results in a reasonable time.

VII. **Alert's module**: The alerts module implementation enables AIOps to send frequent email alerts to registered users informing them about the new set of predictions, a graph of the

predictions for next few hours, the min and the max value that the metric touches and any other details if required.

8. Implementation details

Figure 8.1 shows the implementation and control flow for the Metric Forecasting API. The metric forecasting has two forms on the user interface - for **registering a new metric** and for **fetching available predictions**.

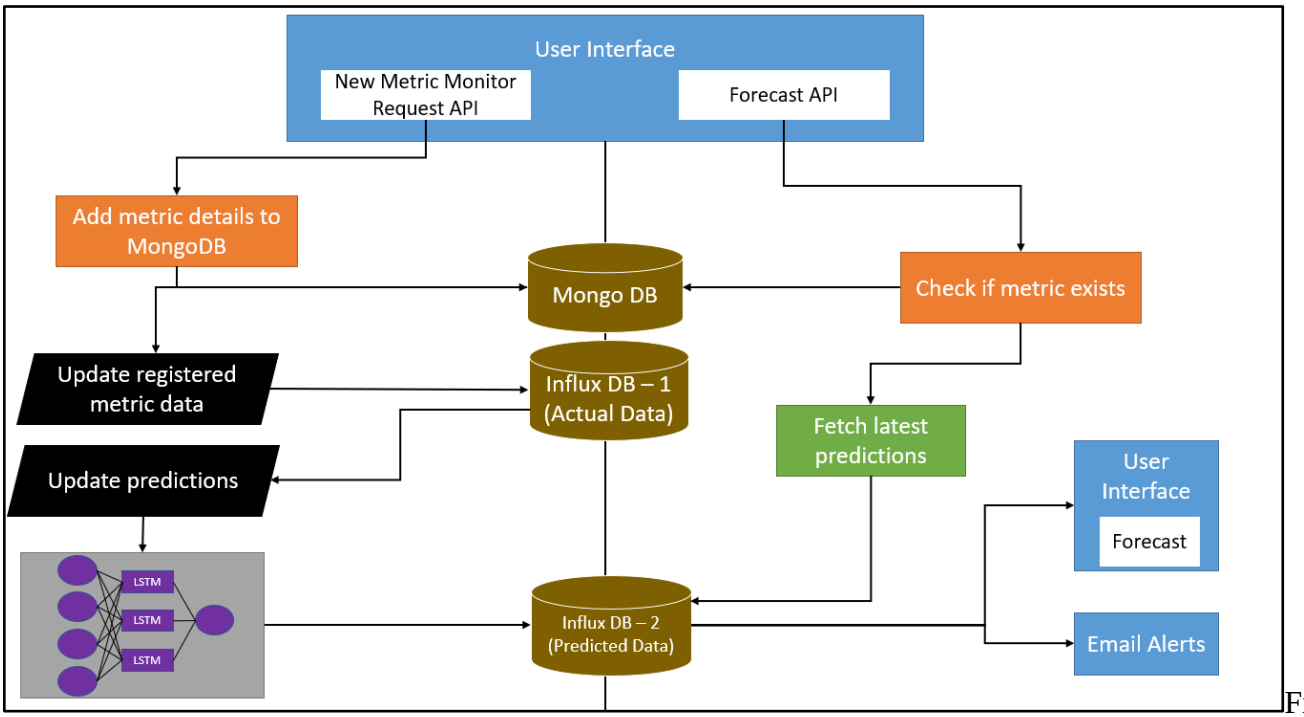

Fig 8.1 - Implementation Metric Forecasting (Backend architecture)

The prediction process itself is isolated and takes place independent of any API, according to the current implementation, **a cron job at 12 midnight runs the forecasting function** that collects the past 48 hours data **to forecast the next 48 hours**. The predictions are stored in Influx DB. At any point if a user requests for the predictions or if email alerts need to be sent to users with the metric forecast graph and data, the predictions are easily obtained from Influx DB and displayed on UI/ sent via email. Therefore, it takes **no processing time to obtain the predictions**.

Now, as mentioned in the previous section, several ML algorithms were analyzed and their performance and accuracy was compared to find the best model for time series forecasting. Below is a brief description of the algorithms used:

I. **Auto Regressive Model (AR)**: The AR model relies on past period values to predict the future values. It is a linear model where the future value is the sum of the past values each multiplied by some weight. An AR(p) model is one where the last p lags are considered to predict the next value.

II. **Moving Average Model (MA)**: The MA model has a similar working like the AR model but it takes past residuals or errors into account to make the future prediction. Works very well when there is low fluctuation in a small-time gap.

III. **Auto Regressive Moving Average Model (ARMA)**: The ARMA model considers the AR as well as MA together to make future predictions. Readily available to use in the statsmodel python package, an ARMA(p,q) is where the last p lags are considered in the AR part and last q lags are considered in the MA part. ARMA does not work well for non-stationary data and hence did not perform well in metric forecasting.

IV. **Auto Regressive Integrated Moving Average Model (ARIMA)**: An ARIMA model overcomes the issue with ARMA regarding the stationarity of data. An ARIMA(p,q,r) is where p lags are considered in the AR part, r lags are considered for the MA part and the entire dataset is differenced by a degree of 'q' to convert it from stationary to non-stationary.

    (To determine the number of lags to be considered in AR, MA, ARMA and ARIMA, various techniques can be used. In this project, autocorrelation plot (ACF) and partial autocorrelation plot (PACF) are used).

V. **LSTM Recurrent Neural Network (LSTM-RNN)**: Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video). The LSTM RNN performed the best out of all the given models.

Another aspect of the implementation was to come up with a logic for predicting values and then **monitoring** with the real values as they come in to understand if the model is really performing well or not. There could be a case where values predicted by the model are far off from the actual values that are also being collected simultaneously.

To implement such a logic to take corrective action or consider retraining the model based on its performance in real time, the idea explained in the below image (Fig 8.2) is considered.

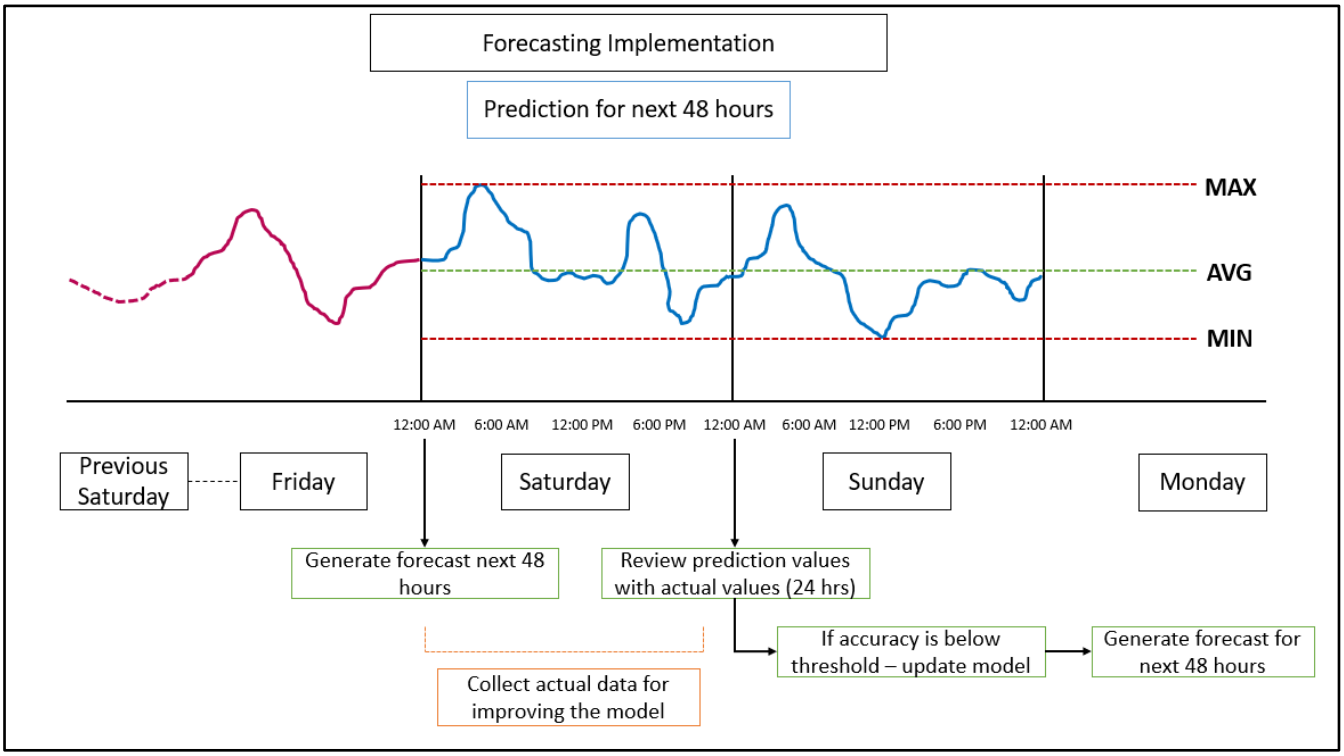

Fig 8.2 - Implementing Metric Forecasting (Model training and review)

The general idea and the current implementation for the same, is to use past data to predict the next 48 hours at any point of time. An important point to note is that making predictions based on a trained model or re-training the model and **making predictions demands high processing power and computation time**. To perform such an operation, a time window had to be figured out where the AIOps service would have relatively less load and would not have too many developers trying to use the service. Midnight 12 was considered as such a time in the current implementation as it can be seen in Fig 8.2.

Once the predictions are made and stored in Influx DB, **the predictions can be reviewed with the real time values** that are being collected by the Datadog agent. A **threshold** can be set by the AIOps administrator to alert the AIOps team if the difference between a particular metric prediction and its corresponding real value is much greater than the threshold. In such a case, the AIOps team can take corrective action. Another solution would be to automate this process and

measure the difference between actual and predicted values at certain intervals and if the differences between the two are significant or much greater than the threshold defined then simply discard the existing predictions and retrain the model to make new predictions based on newly available data.

This was in summary the implementation of the metric forecasting component of AIOps, similar implementations and ideas are being used in the ongoing development of other components like the anomaly detection, outlier detection and automated log analysis.

9. Project result

The following results have been obtained at the current stage of the project:

I. AIOps at its current stage has proved to be an easy to develop and **essential utility for any software development team**. It provides real time insights to developers which allows them to enhance the application, remove bottlenecks or any other inefficiencies and track the application and system metrics of their software.

II. AIOps **reduces third-party dependencies** of a company as usually the third-party apps are made at a very generalized level that might constraint its use in certain scenarios. AIOps on the other hand is tailored to fit the requirements of the developers and the teams using this utility.

III. AIOps is **cost effective** as it requires minimal server space and doesn't use processing power at all times. As stated in section 8, the metric forecasting feature of AIOps is scheduled to run once in 48 hours and the predictions are always fetched from a database which is instant data retrieval with no delay.

IV. As mentioned in section 8, several ML algorithms for time series predictions were compared to find the best model for the metric forecasting feature, below is the result of the comparison. The accuracy considers +1/-1 deviation from actual value. The metric for which the below accuracy is listed is the number of concurrent requests on a Google Cloud Platform (GCP) instance for the ASR module in the South Korea region for Bixby 2.0. The data used was actual production data

Algorithm Name	RMSE	Accuracy
Auto Regressive Model (AR)	0.7534	81%
Moving Average Model (MA)	3.654	73%
Auto Regressive Moving Average (ARMA)	5.832	54%
Auto Regressive Integrated Moving Average (ARIMA)	0.8153	83.75%
LSTM RNN Model	0.809	94%

Table 9.1 - Comparison of time series prediction algorithms

10. Learning outcomes

The Learning opportunities as part of the internship have been plenty. Right from the project planning phase all the way to implementation and deployment, understanding how industry level decisions have to be made was possible only through the exposure of working in such a company. The crucial decisions such as **tradeoffs** between efficiency and performance, which work item to take up first and **how to approach other team members** and senior staff were few of the learning outcomes from the non-technical point of view.

From a technical point of view, learning about **time series prediction** - the type of data (stationary vs non-stationary data), the various types of models, etc. has been one of the important learnings. Another outcome is working with **Influx DB** which is primarily a time series database and one that I have not worked with earlier. Understanding how to store and retrieve data from Influx DB measurements, how to setup buckets with minimum redundancy and performing other database operations has been few of the interesting topics to spend time on. Another technical learning was the two Linux utilities **Mutt and Cron**. Mutt is a command line-based Email client to send and read mails from command line in Unix based systems. It allows sending periodic alerts to registered users. Cron also known as cron job is a time-based job scheduler in Unix that allows running processes (python code as well as bash scripts) at scheduled time making it easier to automate processes and reduce manual intervention.

11. Future scope

AIOps at its current state has the metric forecasting feature completely up and running, along with this the anomaly detection and outlier detection are also towards completion. The future scope of AIOps would include more features that enables developers to get helpful insights to their applications. Some of the features could be:

I. Setting up an AIOps agent to directly monitor the metric and not fetch them from Datadog.
II. Geomaps to visualize app data by locations.
III. Visualizing network communication and flow between client and server.
IV. Root cause analysis in case of application crash or errors.
V. Data driven stories.
VI. IoT agents to monitor IoT devices.

12. Conclusion

The potential of **AIOps - AI/ML enabled Operations** in today's time is immense, any software company will benefit from an AIOps like utility because implementing and deploying a software/application does not complete a developer or team's work with regard to that project, constant monitoring of the application to identify the issues and bottlenecks is important for the continuous growth of the product and to deliver a smooth experience of using the application to the user.

I had a very fruitful learning experience during my internship at Samsung Research Institute, Bangalore. All the team members were always helpful and encouraged me to think outside the box, implement ideas of my own and enjoy the learning experience.

13. Bibliography

I. https://machinelearningmastery.com/time-series-forecasting/

II. https://machinelearningmastery.com/autoregression-models-time-series-forecasting-python/
III. https://machinelearningmastery.com/moving-average-smoothing-for-time-series-forecasting-python/
IV. https://towardsdatascience.com/advanced-time-series-analysis-with-arma-and-arima-a7d9b589ed6d
V. https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/
VI. https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/
VII. https://www.tensorflow.org/tutorials/structured_data/time_series
VIII. https://www.tecmint.com/send-mail-from-command-line-using-mutt-command/
IX. https://opensource.com/article/17/11/how-use-cron-linux
X. https://docs.influxdata.com/influxdb/v2.0/
XI. https://docs.mongodb.com/
XII. https://flask.palletsprojects.com/en/1.1.x/
XIII. https://vuejs.org/v2/guide/
XIV. https://people.duke.edu/~rnau/411diff.htm
XV. https://pymongo.readthedocs.io/en/stable/
XVI. https://docs.datadoghq.com/
XVII. https://help.sumologic.com/Metrics/Metric-Queries-and-Alerts

Along with the above references, a lot of material from Samsung's internal portals was used to get better understanding for the same topics. Due to company policy, those references cannot be cited in this report.