International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 07 | July 2022 www.irjet.net p-ISSN: 2395-0072

# Development of email notification system based on user criteria

**Sai Charan Vikranth Paluri[1], Sowmya Nag K[2]**

*[1]Student, RV College of Engineering, Karnataka India*
*[2]Assistant Professor, RV College of Engineering, Karnataka India*

---

**Abstract -** *One of the most important feature of any customer centric application is notification. Email notification lets users to get notified about any crucial updates via email without having the need of user being logged into the application. Criteria based notification enables users to set certain criteria upon which the notification can be sent via email. Through the analysis, design and implementation of the email notification system for small size user groups for getting notified about the data source change that updates the dashboards, this paper studies how to collect the criteria and notify data without affecting the business system, and proposes a scheme of using a data pipeline to check for the change in data to notify the users. It also compares two methods of implementing the email notification service. The first method utilizes Java based web framework known as Spring boot which is used in building REST APIs and Sendgrid which is a cloud-based SMTP provider. The second method utilizes Azure logic app which is an Azure service to create automated workflows. The end users receive email notification upon a change in the data source indicating the source of data.*

***Key Words*: Azure, Spring boot, SQL;**

## 1. INTRODUCTION

An email notification is a message that is sent to the subscribers to let them know about updates or changes to a website, data or service, such as new features, update to the dashboards, products, or planned website maintenance. It's best to prepare the users of the application in case they need to change their routine in order to accommodate changes to a website or service operation. The potential negative impact of changes will be reduced by sharing news about updates.

Azure functions are a serverless idea in cloud native architecture that enables code deployment and execution without the requirement for server infrastructure, a web server, or any configuration. Functions in Azure can be scaled. When there is more demand for execution, the service is automatically given more resources, and when there is less demand, all excess resources and application instances are automatically terminated.

Azure Logic Apps is a cloud service. When using Logic Apps, a variety of APIs that are accessible as Connectors are easily consumed by the workflow. When the trigger is triggered, these Logic App connectors will carry out the series of operations specified in the workflow. A workflow in an Azure Logic App is a series of steps-define processes or activities. When input is delivered through the managed connectors, the actions will carry out the business processes. Each workflow will begin when the trigger is fired.

Spring Boot is an open source Java-based framework used to create a micro service. It is used to build stand-alone and production ready spring applications. Important layers of a spring boot application are:

- The controller layer which handles all the API requests incoming. Upon hitting the end point, the controller class contains a method of data access object interface.
- The data access object interface contains methods that interact with the database directly. This method further calls the method from the service layer where the core business logic lies.
- The service layer contains the core business logic to handle the implementation service. This layer utilizes third party services and functions to implement the service for business use.

Unit testing of the code is performed using Junit, which is a unit testing framework for Java. It provides an annotation with test so that a method can be written to perform the test if the method is serving the purpose. The service layer method is tested by comparing the response code after calling the service layer method with sample data.

Jacoco is an open source tool which is used to perform code coverage. Code coverage gives the percentage of code covered by running all the test cases and going through the entire code. It also provides a visual representation of the code which is not covered.

## 3. METHODOLOGY

The methodology to build the email notification shown in the Fig. 1 system has some assumptions. The first one being to notify users only about the data source change. The important steps include building data pipeline, Azure function and finally the email service. Microsoft SQL server is used as the database and it's compatible with Azure

platform. The important steps in methodology include:

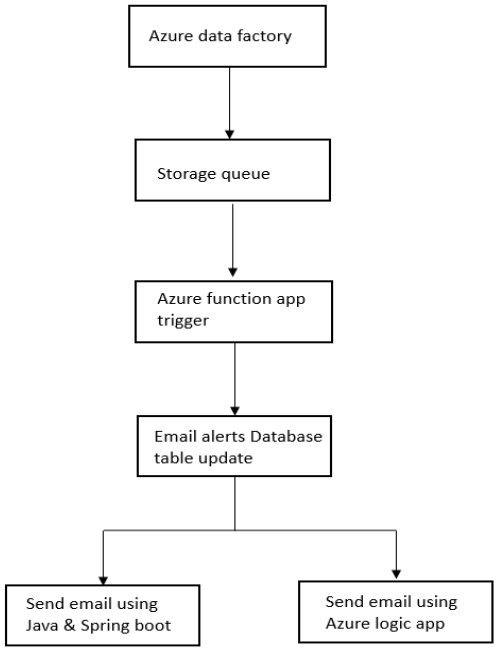


**Fig.1.** Email notification workflow

### 3.1 Building data pipeline

Azure data factory is to build a pipeline that runs and monitors both the types of data sources which populates the dashboards. The pipeline is responsible to capture the change in the data source and put the message into the queue.

This can be achieved by first storing some important parameters/tokens of the pipeline in Azure key vault.

The last step involves posting the message into the storage queue. An additional Web activity is created to put a message onto the queue through a POST API call. The URL is concatenated and contains various parameters like key variable, queue name and the account in which the pipeline is created.

### 3.2 Azure function

Azure functions are server less functions which runs without the need of having any infrastructure in place. Azure function app is created using Java as run time environment. The function app basically works on a principle of queue trigger. The storage queue which is configured in the first step is the queue which the function app monitors. The purpose of the function app is to update the email alerts database with all individual alerts corresponding to all the users whose notification criteria matches.

The function app first fetches out all the details like alert notification criteria id, unique id of the user, data source id and their Boolean values if the user has set them to true. The function app then loops through all the results and constructs the body and subject of the email and inserts them into the email alerts table.

### 3.3 Email service using Java and Spring boot

The implementation of the email service is implemented using Java based framework called Spring boot. The controller layer is responsible for handling the various API requests incoming. POST **/api/emails/id/send** This is a post request to send an email to all the email ids in that particular record with corresponding subject and body. The method takes the string containing the email ids to which the email has to be sent, subject and body of the email. The method constructs the message by setting the from email, subject and the body. Personalization object is created and all the email ids are added to this. SendGrid API key is used to establish the connection and to send the mail. Unit testing for the service layer is done using a unit testing framework called Junit.

### 3.4 Email service using Azure logic app

Logic app is used to create an automated workflow. The workflow of the logic app developed for the email service is shown in the Fig. 2. The first step of the logic app checks the email alerts table for every one second. In the second step a row is fetched from user table based on user ID obtained from the new alert row. In the next step Sendgrid is used to send the email to the corresponding user email. The body and subject of the email are obtained from new entry of the email alerts table. In the last step, the is new status of the email alert in the email alerts table is updated to false indicating that the email has been sent and is no longer a new email alert.

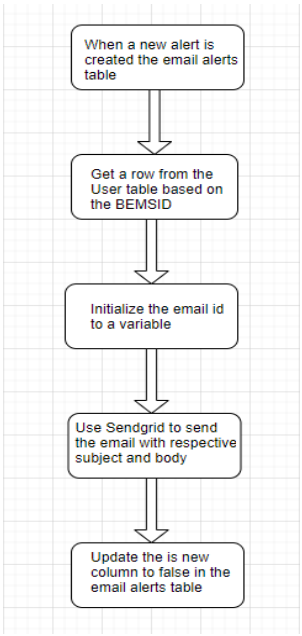


**Fig.2.** Logic app workflow

## 4. Result

The result is briefly divided into four parts. The first one being the result of Azure function app updating the database table, second one being the result of email notification using Spring boot, third one being that of the logic app and the last result comparing the performance and benefits of the two methods of creating the email notification.

### 4.1 Azure function app result

The azure function app which is deployed gets triggered and runs when there's a new message in the storage queue. The JSON message contains the notification criteria id created by user and the notification criteria name.

Once the function app gets executed successfully, it updates the email alerts database table with email alerts each corresponding to a single user. The database table gets updated with three new alerts after the function app gets executed since the message in the queue matches with the notification criteria created by three users.

### 4.2 Email notification using Java and Spring boot

The first step to obtain the results using Spring boot is to update the email alerts database with new alert. The table contains a unique id of the alert, subject of the email, body of the email and the string containing the list of email ids separated by comma. The email is then sent with the HTML specific template as shown in figure 3. The HTML can be embedded in the body of the email.

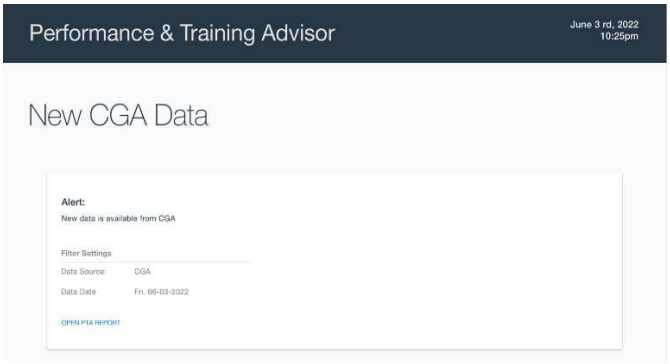


**Fig.3.** Email notification template

### 4.3 Email notification using Azure logic app

The logic app performs an automated workflow whenever there's a new entry in the email alerts table. The new row in the email alerts table doesn't have the email of the user to which the notification has to be sent.

The logic app reads the row from the email alerts table and finds out the email from user table using the user specific ID. The email is then sent with a plain text indicating the change in source of the data to the corresponding user. SendGrid in logic app doesn't support HTML embedding as it only supports plain text. The number of succeeded runs of the logic app workflow is same as the number of users to whom the email has sent. As the queue message corresponds to three users who has subscribed to the notification, the three runs get succeeded. The response time of the request is captured shows that the request has taken 166 ms to execute and send the email to all the users.

### 4.4 Comparative analysis

The table shows the comparison of the response time and the memory of the email service of Java/Spring boot and Azure logic app.

The email service using Azure logic app takes 310 ms to execute whereas using Java and Spring boot takes only 166 ms. As Azure logic app runs on cloud premise it doesn't take any memory even for high volume of email notifications

The email service using Azure logic app doesn't support embedding HTML template into the email body whereas using Java and Spring boot supports embedding HTML template into the email body.

**Table 1** Comparison of email service methods

| Email service method | Execution time | Memory |
|---|---|---|
| Spring boot and Sendgrid | 166 ms | 123 bytes |
| Azure logic app | 310 ms | 0 bytes |

## 5. Conclusion

Email notification in particular comes very handy when the application users want to get notified about some changes in their data without opening the application.

The system architecture utilizes Azure storage queue to store the messages from the pipeline constantly running and updating the storage queue upon the change in the data source that feeds the dashboards in the application. Azure storage queue proves to be a great option because of it's on premise availability and its easy integration with other Azure services. The Azure function which is used to monitor the storage queue and update the email alerts database table with alerts corresponding to users who has subscribed to receive that particular notification provided smaller execution time and memory and provided easy integration with the storage queue and the database.

Sending email using Spring boot in the back end provided lesser execution time compared to logic app but has consumed more memory than logic app. Thus, the use of Spring boot provided more options to include with complex HTML templates and lesser execution time. Logic app should be used for lesser critical applications and for easier implementation. Overall using Spring boot in the back end along with Sendgrid service provided better performance as execution time is lesser and it provided an option to include custom HTML templates.

## REFERENCES

1] X. Zhang, Y. Jiang, J. Miao, and Q. Li, "A data-driven analysis method for aircraft flight performance comparison," in 2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT), IEEE, 2021, pp. 328–331.

[2] W. Hu, H. Xie, M. Nakas, W. Shi, and M. Wang, "Power bi for impacts analysis on cost of living caused by industry prevalence in smart cities," in 2019 3rd International Conference on Smart Grid and Smart Cities (ICSGSC), IEEE, 2019, pp. 134–139

[3] R. Pukala, S. Hlibko, N. Vnukova, and O. Korvat, "Power bi in ict for monitoring of insurance activity based on indicators of insurance portfolios," in 2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T), IEEE, 2020, pp. 393–401.

[4] K. Lappanitchayakul, "Development of email and sms based notification system to detect abnormal network conditions: A case study of faculty of business administration, rajamangala university of technology phra nakhon, thailand," in 2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), IEEE, vol. 3, 2018, pp. 98–105.

[5] S. H. binti Husin, Y. binti Yusop, A. H. bin Hamidon, et al., "Real time mailbox alert system via sms or email," in 2007 Asia-Pacific Conference on Applied Electromagnetics, IEEE, 2007, pp. 1–4.

[6] R. Poenaru, "Implementation of an email-based alert system for large-scale system resources," in 2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet), IEEE, 2021, pp. 1–6.

[7] N.Neelima, "Email alerts on whatsapp," in 2021 JETIR July 2021, Volume 8, Issue 7, JETIR, 2021, pp. 1–4.

[8] D. Jackson and G. Clynch, "An investigation of the impact of language runtime on the performance and cost of serverless functions," in 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion), IEEE, 2018, pp. 154–160.

[9] A. Seˇskar, B. Milaˇsinovi´c, and K. Fertalj, "Workflow for image categorization using ˇ cognitive computing services," in 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), IEEE, 2018, pp. 1551–1556.

[10] McGrath, Garrett, and Paul R. Brenner. "Serverless computing: Design, implementation, and performance." 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). IEEE, 2017.