SARATH K S

| Palakkad, Kerala | +91 6282938954 | [sarathsankarnarayan96@gmail.com](mailto:sarathsankarnarayan96@gmail.com) |
| --- |

# Objective

| Certified MuleSoft Developer, looking forward to work in a reputed firm that will enable to use my strong organizational skills, award-winning educational background, and ability to work well with people. |
| --- |

# EDUCATION

| 2014-2018 | University Of Calicut  B.Tech Computer Science and Engineering  8.71 CGPA |
| --- | --- |
| 2012-2014 | Karimpuzha Higher Secondary School  Computer Science  91.6% |

# Experience

| 2018-present | MuleSoft Developer  Accenture Solutions Pvt.Ltd  Bengaluru, India  Design, develop and deploy Integration solutions using Anypoint Studio, Anypoint Platform and API led connectivity approach |
| --- | --- |

# interests

| * Middleware * API Design * Automation * Anypoint Connectors * CloudHub |
| --- |

# TECHNICAL sKILLS

| * Mule ESB * API design using RAML * Dataweave * XML * JSON * Anypoint Platform * CI/CD using Jenkins and Maven * Core Java * MySQL |
| --- |

# ACHIEVEMENTS & AWARDS

| * MuleSoft Certified Developer – Level 1 (Mule 4) * MuleSoft Certified API Design Associate (RAML 1.0) * University 5th Rank Holder * Accenture Certificate Of Excellence - Individual contribution for implementing future skill and automation * Actively participated in MuleSoft Hackathon 2020 |
| --- |

# PERSONAL STRENGTHS

| * Hardworking * Flexible * Problem Solving * Quick Learner |
| --- |

# PERSONAL Profile

| * Date Of Birth : 19-06-1996 * Marital Status : Single * Nationality : Indian * Known Languages : English, Malayalam, Tamil, Hindi * Hobbies : Singing, Music Production |
| --- |

# PROJECTS

* **Integration of SAP and Coupa**

Coupa is a SaaS application that manages purchases, invoices and expense in real time. A scheduler is placed in the source that executes the flow every 5 minutes. Mule receives payload as XML from the coupa via a REST API, which expects an API key in headers to respond with the data. The application deals with two types of data, master data and transactional data. Master data is the data that is created first (users, plant orders, suppliers, approvals etc.) and transactional data, the data that flows from coupa to SAP and vice versa.

Global configurations are created in a separate Configuration file to improve reusability. Common flows such as logging, sending success or failure response to coupa are also designed in a separate configuration file to improve reusability and reduce complexity. Property placeholder is used to remove the usage of hardcoding values. Truststore and Keystore are used to connect to HTTPS enabled Coupa API. Developed in Mule 3.8.5. Async scope is used to store the last ID, whenever the flow gets executed. For each scope is used to process each purchase orders or goods receipt or invoices being received. Connectors used are HTTP, SAP, choice router, couple of filters, Transform Message and Loggers.

* **Integration of Google Assistant and MySQL database**

The project is developed as part of the MuleSoft Hackathon. It is a doctor booking system where users can get an appointment with a doctor by providing voice commands through Google Assistant. Mule receives an HTTP request from Google Assistant. This request contains an Event Handler, which is passed through some series of choice block that does the operation. Google Assistant calls the Mule application through a WebHook. Mule receives the date and time from the user, which is then passed through some validations to check whether the slot is already booked. This is achieved by a select operation from the database and using a filter operation in data weave. If the slot is booked, then Mule Sends back an error message. If the slot is available, then the user data is inserted into the database and an email send operation is triggered to the user’s email address with the appointment details.

Developed in Mule 4.3. Security is achieved by using an API proxy, where Google Assistant calls the proxy endpoint instead of the actual endpoint. Connectors used are HTTP, database, email, object store (to store the user information temporarily), Transform Message to transform the data and loggers to log the execution of the flow.

* **Integration of Smart Barley, Eleaf and Banqu**

Eleaf and Banqu are two web services that contains the personal details and account details of the barley farmers respectively. Smart Barley is an azure SQL database. Eleaf sends the farmer and region details to smart barley database. Banqu sends the account information to smart barley database. Mule receives data from Eleaf and Banqu through an HTTPS call. Both Eleaf and Banqu expects a Bearer token that needs to be passed as Headers to get the data. A scheduler is placed on Eleaf and Banqu to Smart Barley flow that gets executed every 7 days. The data received from Eleaf and Banqu are XML data and this data is converted to MySQL queries using Java.

Any User who provides a valid client id and client secret can access the data present in the Smart Barley database through a REST API. API is designed using RAML to connect user with the Smart Barley database. API is designed using Design Center and is maintained using API Manager. Used APIKIT Router to automatically route the incoming request to the corresponding flow. Autodiscovery is established to achieve a synchronization between the application in Runtime Manager and API Manager. Client id enforcement, Rate Limiting policy and API Proxy is established to improve the security.

A weekly report will also be generated and saved into the database. Used scatter-gather that generates the report of each Brewing Factory in parallel. The report is generated in CSV format. Developed in Mule 3.9.4. Connectors used are Poll, Database, Java, Scatter-Gather, Azure, and Logger for logging the execution.

* **Integration of SharpOps and Payspace**

SharpOps and PaySpace are third party payroll related systems. SharpOps is an SAP system and PaySpace is a web app. Mule acts as an interface between two systems. SharpOps sends employee details as an IDOC to mule. This data is then send to payspace via SOAP web service call. SAP connection is the source that calls a BAPI function through a Transactional RFC. SharpOps sends three types of data, employee data, tax profile data and position data. A choice router is placed in Mule flow to validate the type of data being received. There are few other validations like New Hire, Termination, and position change (when an employee gets promoted).

Developed in Mule 3.9.4. Connectors used are SAP, Web Service consumer, Choice Router, Transform Message to create SOAP structure and Loggers to log the payload and execution.

* **Integration of Service Now, Zoom and SAP**

This project is currently under development. Users raise for PR requests in SAP. This request needs to be approved by someone in the organization. Each time a user raises for the request, the approver should login to the SAP system, which lacks mobility. To eliminate this problem, Zoom and SAP was integrated. When the user raises the request, mule receives this request as XML and sends it to the zoom chatbot. Payload is send from Mule to chatbot through an HTTPS request. This chatbot forwards this message to the desired approver. Approver can approve or reject the request just by typing a message. Based on the response from the approver, the request is approved or rejected in SAP.

When the user request for the PR approval, one incident is automatically created in Service Now. When the request is approved by the approver, incident is closed and when rejected, the reason provided by the approver will be updated in the Service Now. Integration between Service Now and SAP makes it easy to keep track of which all users have requested for the approval.

Runtime Server is Mule 4.3. Connectors used are SAP, Service Now, HTTP, Error Handlers, loggers and async scope for executing two flows (creating an incident in service now and requesting for the approval) in parallel.

* **Integration of SAP FI and Blackline**

SAP FI is a system that contains finance and accounting information. Blackline is a third party system that automates complex, manual and repetitive accounting processes and enables companies to move beyond the legacy record-to-report process. Blackline system contains a couple of SFTP folder where mule places the file. Mule receives data from SAP in the form of XML through Synchronous RFC call. A choice validation is designed to validate the type of file received from SAP system. This validation is done based on the file name. Based on file type, data is placed in the corresponding SFTP folder. In case if there are any exceptions (SFTP is not available), an email notification is triggered with the details of the file and the cause of the error.

Developed in Mule 4.2. Connectors used are SAP, SFTP, email, Choice Router and Loggers to log the execution of the flow.