



Salesforce Event-Driven Architecture Hands-on Workshop: Student Workbook

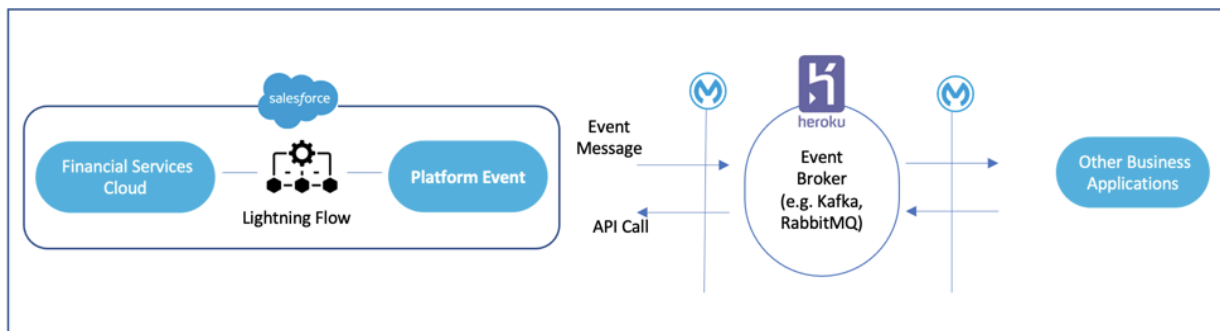
Fast track tutorial to integrating event-driven systems using Salesforce Platform. Hosted by Fins Cabin. [@Fakhruddin Faizi](#).

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Introduction

An event-driven architecture is composed of decoupled systems that run in response to events.

In this tutorial, you use the Salesforce Platform technologies such as **Lightning Flow**, **Platform Events** & **MuleSoft** to build an end-to-end architecture that enables applications (e.g. **Financial Services Cloud** and a **custom application on Heroku**) to communicate via events in a loosely coupled design.



WHAT YOU WILL LEARN

How to create and deploy an end-to-end event-driven architecture by leveraging the following systems:

- Salesforce Lightning
 - Configure UI Financial Services Cloud (Lightning App Builder, Lightning Component)
 - Create Lightning Flow
 - Create Platform Event
 - Monitor Platform Events
- MuleSoft

- Create MuleSoft Process Flows
- Transform your message payload using MuleSoft DataWeave
- Heroku
 - Deploy a database (MySQL) on Heroku
 - Deploy an event broker (RabbitMQ) on Heroku
 - Deploy a custom application on Heroku which acts as both event producer and event consumer.

> Tools & Systems Used in this Workshop

PREREQUISITES

- **Laptop Connected to the Internet:** To complete this workshop, all you need is a modern browser and a laptop with a connection to the Internet.
- **Browser Requirements:** Most recent versions of Chrome work best. Other browsers may also work (e.g. Microsoft Edge) however you may not be able to do some modules (e.g. GitHub requires chrome).
- **Knowledge Level:** No prior knowledge of Salesforce, MuleSoft, or Heroku is required. Very basic knowledge of integration best practices is assumed. However, we assume that you have attended the “*Event-Driven Architecture for Financial Services*” event in which we have discussed the eventing concepts & systems in depth.

The workshop does assume that you enjoy configuring & troubleshooting enterprise systems and like to learn by exploring and experimenting.

> How to use this Guide (Student Workbook)

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Setup

Sign up for your free trial instances.

- Financial Services Cloud (FSC) trial Org: <https://www.salesforce.com/form/signup/financial-services-cloud-trial.jsp>
- Heroku trial instance: <https://signup.heroku.com/>
 - In the next module during the deployment of the starter app, you will be prompted to verify your free Heroku account through a credit card.
 - Credit card verification is a security requirement for the use of third-party add-ons and custom domains. We are using free third-party add-ons (MySQL & RabbitMQ) in our workshop. You will not be billed in any way for using free add-on plans even if you have your credit card information on file. Please see this note for additional information: <https://help.heroku.com/WZIGJX28/why-do-i-need-to-have-a-verified-account-for-a-free-add-on-or-custom-domain>
- MuleSoft AnyPoint Platform trial instance: <https://anypoint.mulesoft.com/login/signup>
- GitHub signup: <https://github.com>

Additional Information: Module 1: Setup (Supplementary Information): 📄 **Salesforce Event-Driven Architecture Hands-on Workshop: Workbook Supplement**

> What You Have Accomplished

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Configure

Financial Services Cloud (FSC) Org Configuration

We assume you have completed Setup and are now logged into your FSC Org.

Summary of what we shall do in this section: In this section, we shall install a starter configuration in our new FSC Org. This will install some pre-defined metadata (including a screen flow). We shall then configure the Account page with this screen flow.

- Click on this “unmanaged package” link to install a starter configuration in your FSC Org (Select “Install for All Users,” select defaults for everything else: <https://login.salesforce.com/packaging/installPackage.apexp?p0=04t5w000005dk6m>)
 - This configuration installs pre-built business process definitions created with Lightning Flow and custom Platform Event object in your Org.
- Navigate to the account tab in FSC, click on the Account tab in the navigation bar, and then on the “All Accounts” list. Click on any account to open up an account.
 - Navigate to “Setup” (top right) → “Edit Page.”
 - Clicking on ‘Edit Page’ opens up the “Lightning App Builder” which allows you to customize your page using pre-built Lightning Components which are available on your left sidebar.
 - Now you are in the edit mode in the account object’s page layout.
- In the left sidebar search for “Flow” and you will see the Flow Lightning Component. Drag and drop the “Flow” component above the “Activity” in the upper right corner of your page layout. The activity component which is currently there will move directly underneath.
 - When you click to select this component you have just now added in the page layout - you will see a list of flows in the right sidebar available - Select the Account Address Change Flow. Now scroll down in this same sidebar and click the checkbox (“Pass record ID into this variable”) below the flow variable “v_RecordID”.
 - Now click “**SAVE**” on the top right of the page, then click on the “**Activation**” button next to the Save.
 - In the next page, click on “**Assign as Org Default**,” and on the next screen, click on “**Desktop & Phone**”.
 - You can now click on the left arrow in the top left of your page and it will take you back to the account page.
 - You should now see the Account Address Change Flow embedded in the top right corner of your account page when you click on any account.

Screenshots: Financial Services Cloud Configuration: 📖 Workbook Supplement

> What You Have Accomplished

Heroku Configuration


We assume you have completed Setup and are now logged into your Heroku instance.

Summary of what we shall do in this section: In this section, you shall install a starter application in your newly created Heroku trial Org. We shall also create an event queue in the RabbitMQ that gets installed in this section.

- Install the finscabin app
 - Click on the “finscabin” Heroku Button on this GitHub page (scroll to the bottom of the

page): <https://github.com/lightningexperience/finscabin>

- Clicking on the Heroku Button brings up a dialog box in your Heroku instance asking you to choose an 'App Name.' Choose any unique name and leave all the other defaults.
- Navigate to the bottom of your page and in a couple of minutes, you will see the process completion message. Click on "Manage Your App" to open up your newly installed app.
- Configure and bind a queue in RabbitMQ
 - Navigate to Resources → CloudAMQP. This opens up the CloudAMQP console. Click on RabbitMQ Manager on top left.
 - Create a queue and name it **eda01** - accept all other defaults.
 - Go to Exchange → **amq.direct**. This brings you the page for this exchange. Scroll down and put in the name of your queue (eda01) where it says "Add binding from this exchange" "to queue," and press on Bind. Accept all other defaults.

Screenshots: Heroku Configuration::  Workbook Supplement

> What You Have Accomplished

MuleSoft Configuration

We assume you have completed Setup and are now logged into your MuleSoft Anypoint Cloud.

Summary of what we shall do in this section: In this section, we shall import MuleSoft process definitions (aka Mule flows). We shall then configure connection information to connect our different systems together using MuleSoft.

- Download the jar file from the following link:
 - <https://github.com/lightningexperience/finscabin/blob/main/resources/finscabin-sf-to-db-1.0.0-mule-application-template.jar>
 - <https://github.com/lightningexperience/finscabin/blob/main/resources/finscabin-02-db-sf-1.0.0-mule-application-template.jar>

The .jar files are "bundles of logic" or templates that simplify MuleSoft. We have created some basic transformations and connector configurations to give you a fast start in the workshop. Once you deploy these .jar files, you will be able to inspect and examine all these configurations yourself and optionally create your own templates.

- Navigate to Design Center (link in the top left tab). This will take you to the Design Center Project page.
 - Click on "Create New" → "Import from File". Locate the first file you downloaded ([finscabin-sf-to-db-1.0.0-mule-application-template.jar](#)) and import it. Choose the name 'sf to db' and press import.
 - Again - Click on "Create New" → "Import from File". Locate the second file you have downloaded ([finscabin-02-db-sf-1.0.0-mule-application-template.jar](#)) and import it. Choose the name 'db to sf' and press import.
- Configure connectivity information in each of the two newly imported projects ("sf to db" and "db to sf")
 - Open each element in the project (click on the three vertical dots on the right side of each element) and configure appropriate connectivity information.
 - Press **Test** and then after you get a success message, press **Save to store this connection information in that element**. Then press **Save** again in the next screen to make the connection information active for that element.
 - Repeat the above steps with appropriate connectivity values for: **"sf to db"** project (Salesforce Connector, AMQP Connector (Publish), AMQP Listener, and Database Connector) and **"db to sf"** project (Database Connector/Listener, AMQP Connector, AMQP Listener, Salesforce Connector)

- The connection information for Salesforce Connector is the username password of your FSC Org
- The connection information for your AMQP Connector is the value you get from CloudAMQP installed in your Heroku app by navigating to the page via - Resources→CloudAMQP. For port, use the default RabbitMQ port number (5672).
- For database connectivity information - go to your Heroku app - and then to Resources - JawsDB MySQL.
- It's possible, you may have to save the information without testing. After you have saved all the connection information, you can go back and test connection information in each element and press save again to make sure the connection information is active.

Screenshots: MuleSoft Configuration:: 📄 Workbook Supplement

> What You Have Accomplished

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Explore

Let's test & explore the architecture we have just now deployed.

- **Account Address Change Flow:** Keeping an external MDM database in sync with account address changes in Salesforce is a common process in financial institutions. In our workshop, for example, we sync Salesforce changes to MySQL database via MuleSoft and RabbitMQ.
 - In the MuleSoft Design Center, open up the project - "sf to db" and then press Test on top right of the project page. Keep this tab open.
 - Now in a separate tab, go to your FSC Org and walk through the screens of your Account Address Change flow to change the billing address of an account. After you click Finish on the flow, you can go back and check the MuleSoft tab and you should see a tick mark on each box in the flow confirming that each step of the flow was successful.
 - Confirm that the address change has been synced from Salesforce to MySQL by going to your Heroku app. Click on **"Open app"** in your Heroku app page. This will open up the FinsCabin app and then click on "Customers" in the app. This will show the updated value of the billing address that you had updated in Salesforce.
 - Screenshots: Account Address Change: 📄 Workbook Supplement
- **Lead Transmission:** The leads and referrals captured at various company events need to be synchronized in real-time to Salesforce so the sales teams can immediately act on the hot leads. In our workshop, for example, we sync changes from MySQL database to Salesforce via MuleSoft & RabbitMQ
 - Go to the MuleSoft Design Center and click on your project "db to sf" and then click on "Test" if the project does not start by itself. Keep this tab open.
 - In another tab go to your FinsCabin Heroku app and click on the Leads tab. Enter any email address, first name, and last name and click on the plus sign to save it in your MySQL database. You can click on the refresh button (next to the plus sign) to check the newly entered lead.
 - Confirm that the lead has been transmitted from the Heroku app to Salesforce by going to Leads and Referrals tab in your Financial Services Cloud (FSC) Org and click on "Today's Leads and Referrals".
 - If you don't see the newly created Lead in FSC, then check your MuleSoft window. Is there a tick mark against each step in the flow?
 - **Screenshots:** : 📄 Workbook Supplement

> What You Have Accomplished

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Enhance

> The Big Picture: What You Have Accomplished So Far

What are some ‘enhancements’ that can be built into the event-driven architecture you have just now deployed?

Here are some examples.

- **Change Data Capture (CDC):** For outbound event streaming from a Salesforce Org consider using Change Data Capture instead of custom Platform Events. You can think of Change Data Capture as a special-purpose ‘pre-built standard’ Platform Event construct for synchronizing external systems with Salesforce. Change Data Capture automatically captures the changes in the objects and streams them out without requiring process trigger flows.
- **Complex Event Processing (CEP):** How will you adapt event-driven architecture for complex event processing use cases in your company? In simple event streaming (as in our workshop examples), event consumers act on a single stream of data arriving in chronological order. In complex event processing, data from multiple streams in different time-order is combined to draw inferences.
 - As an example, in financial institutions for Fraud Detection, you can identify fraudulent transactions by tracking real-time events against various patterns. As yet another example, Salesforce Event Monitoring Threat Detection correlates data from multiple real-time event streams to identify anomalous behavior.
- **Fan-out:** Let’s consider another use case. Say a financial institution has three Orgs - Commercial, Wealth and Retail.
 - A life moment happens with your small business owner - the client is getting married.
 - A referral needs to be passed to Retail bank org for, say mortgage and other loans to help the client at this juncture.
 - A referral also needs to go to Wealth org to help the customer rethink his/her financial goals.
 - How would you implement this **fan-out of referrals** using the architecture you have just now deployed?

> Thinking about Eventing

[Main Menu](#) | [Introduction](#) | [Setup](#) | [Configure](#) | [Explore](#) | [Enhance](#) | [Rearchitect](#) | [Resources](#)

Rearchitect

Congratulations! If you have reached till here then you can call yourself an Eventing Blackbelt. You already know more than most experts about Eventing & Integration! Now let’s think about how you can further build on your knowledge.

- **Integrations:** How would you re-architect your integrations to be truly future-proof with a loosely coupled API Network?
- **Microservices Development:** How would your eventing architecture support development processes (e.g. microservices development)?
- **Business Agility:** How would your event-driven architecture become meaningful for your business?

> Where to Go From Here

Resources

Supplements to EDA Student Workbook

- Common Errors & Troubleshooting: Event-Driven Architecture Workshop Common Errors
- Workshop Screenshots: Workbook Supplement: Screenshots
- How to do a similar eventing workshop with Kafka? Integrating Salesforce (Platform Events/Lightning Flow) with Kafka and MySQL: Hands-on Workshop

MuleSoft: Deep Dive Into APIs

- Examine how you can evolve what you learned in the workshop to an enterprise-scale architecture: API Led Connectivity White Paper: <https://www.mulesoft.com/lp/whitepaper/api/api-led-connectivity>
- API Led Connectivity with MuleSoft Trail: <https://trailhead.salesforce.com/users/mulesofttraining/trailmixes/blaze-your-integration>
- Enhance the flows you have created with MuleSoft Cloud Flow Designer using the MuleSoft Anypoint Studio desktop application. Download Anypoint Studio: <https://www.mulesoft.com/lp/dl/studio>

Eventing in Salesforce: Best Practices & Customer Examples

- Event-driven architecture with Salesforce Platform Events: Event-Driven App Architecture On The Customer 360 Platform: <https://developer.salesforce.com/blogs/2020/04/event-driven-app-architecture-on-the-customer-360-platform.html>
- How are other financial services customers using Salesforce Platform Integration tools: Powering Digital Experiences with Data Integration Utilities: <https://developer.salesforce.com/blogs/2020/05/powering-digital-experiences-with-data-integration-utilities.html>

Visualize Salesforce Platform Events

- Streaming Monitor - This is a really great and free tool that can also be used to monitor other streams such as Real-Time Event Monitoring activity streams. You need to have MyDomain enabled in your Salesforce Org - if you don't have MyDomain enabled the custom Lightning Components that this app uses can not render: <https://appexchange.salesforce.com/appxListingDetail?listingId=a0N3A00000FYEEWUA5>
- Platform Events Simulator & Stream Visualizer: <https://appexchange.salesforce.com/appxListingDetail?listingId=a0N3A00000EcrR3UAJ>

Further Reading on Event-Driven Architecture & Integration

- Reliability Pattern in a MuleSoft API Network: <https://docs.mulesoft.com/mule-runtime/4.3/reliability-patterns>
- Transaction Management in MuleSoft: <https://docs.mulesoft.com/mule-runtime/4.3/transaction-management>
- Microservices: SHIFT Commerce's Journey: Deconstructing Monolithic Applications into Services: <https://blog.heroku.com/monolithic-applications-into-services>

Eventing Considerations

- Robust Usage of Apache Kafka on Heroku: <https://devcenter.heroku.com/articles/robust-kafka>

Thank You!