Meeting Summary: Zuora + AI Integration Project

Overview

This meeting was about creating an **AI-powered assistant** that helps developers easily integrate with **Zuora**, a subscription billing platform. The goal is to make it super simple for engineers to understand what APIs to call, what data (JSON payloads) to send, and even give them working code samples they can use right away.

In short: they want to build an **"Integration API Advisor"** — an AI tool that explains *how* to do something with Zuora's APIs and also *does* most of the technical work (like writing JSON or generating SDKs).

What is Zuora?

Zuora is a platform that companies use to manage their **subscription-based business**. It handles things like: - Creating and managing **products** (what's being sold) - Setting **pricing plans** (monthly, yearly, one-time, etc.) - Managing **subscriptions** (who's subscribed and what they pay for) - Handling **billing, invoices, and payments**

For example, if a company sells an online service, Zuora helps manage how customers subscribe, pay monthly, and renew automatically.

What the Team is Building

The team wants to make a smart tool that helps engineers interact with Zuora's APIs without having to manually read through hundreds of pages of documentation.

Here's what the **Integration API Advisor** will do:

1. **Understand Developer Questions**
   Developers can ask questions like:
   → "How do I create a subscription that has a one-time fee and a recurring charge?"

2. **Give Step-by-Step Guidance**
   The assistant will respond with exactly what needs to be done, such as:

3. Create a product
4. Create a rate plan
5. Add one-time and recurring charges

6. Create the subscription (or use the Orders API to do it all at once)

7. **Show Example API Calls**
   It'll show the actual API endpoints (like `/v1/orders`) and example **JSON payloads** developers can use.

8. **Generate Downloadable Resources**
   The AI will also create **Postman collections** (ready-to-test API requests) and even generate **SDKs** (small code libraries) in languages like Python or Java.

In short, it's like ChatGPT for Zuora APIs — but it also gives you working code.

Why This Is Needed

Developers often struggle to understand how Zuora's APIs work together. For example: - They might create a product and think they're done. - But in reality, they also need to create a **rate plan**, **charges**, and **subscriptions** — in the correct order.

The AI assistant will prevent those mistakes by clearly explaining the full flow and giving the right examples automatically.

Example Use Case (From the Meeting)

Let's say a developer asks:

> "I want to create a subscription that includes a one-time setup fee and a recurring monthly charge."

The assistant should: 1. Explain that Zuora needs several steps: - Create the product - Create a rate plan under it - Add two charges (one-time and recurring) - Create a subscription (or an order that includes all those details)

1. Show which Zuora APIs to use (for example: `/v1/catalog/products`, `/v1/orders`).

2. Display the actual JSON payloads (the structured data sent to Zuora's API).

3. Offer a ready-to-use Postman collection or SDK.

How the Team Plans to Build It

They discussed two ways to make the AI assistant learn:

1. **Web Scraping Approach**
2. The assistant searches Zuora's online documentation and learns from it directly.
3. Advantage: Always uses the latest information.

4. Disadvantage: Messy, sometimes inconsistent.

5. **Knowledge Base with Examples (Better Option)**

6. The team already has **Postman collections** and **sample JSON examples**.
7. These can be stored in a **vector database** (like ChromaDB), which lets the AI instantly find and reuse real examples.
8. Advantage: Cleaner, faster, and more reliable.

They'll probably start with this second approach.

Tools and Technologies Mentioned

- **Zuora Orders API** – lets you create accounts, subscriptions, and charges all in one go (instead of multiple separate calls).
- **ChromaDB** – used to store and search large amounts of text and JSON examples efficiently.
- **Postman Collections** – sets of API calls used for testing; can also generate SDKs (code libraries) from them.
- **SDK Generators** – tools that can turn API examples into working Python or Java code automatically.
- **GitHub/GitLab** – will be used to store all the generated SDKs and code examples.

The Plan Moving Forward

1. **Gather all existing API examples** (Postman collections, JSONs, SDKs).
2. **Upload them to a central knowledge base** (ChromaDB or similar).
3. **Train the AI assistant** to understand how each Zuora API works and when to use it.
4. **Test it** by asking common developer questions and checking the results.
5. **Improve accuracy** by comparing answers with Zuora's live docs.

Big Picture

The company wants to make API integration so easy that any engineer (even new hires) can ask the AI assistant a question and get a full working solution — code included.

This project is part of a broader effort to make **developer experience smarter**, faster, and AI-driven. Eventually, this could expand beyond Zuora — to any API integration the company supports.

Summary in One Sentence

The meeting was about creating an AI assistant that understands Zuora's APIs, explains them clearly, and generates ready-to-use examples, so developers can integrate faster without reading endless documentation.