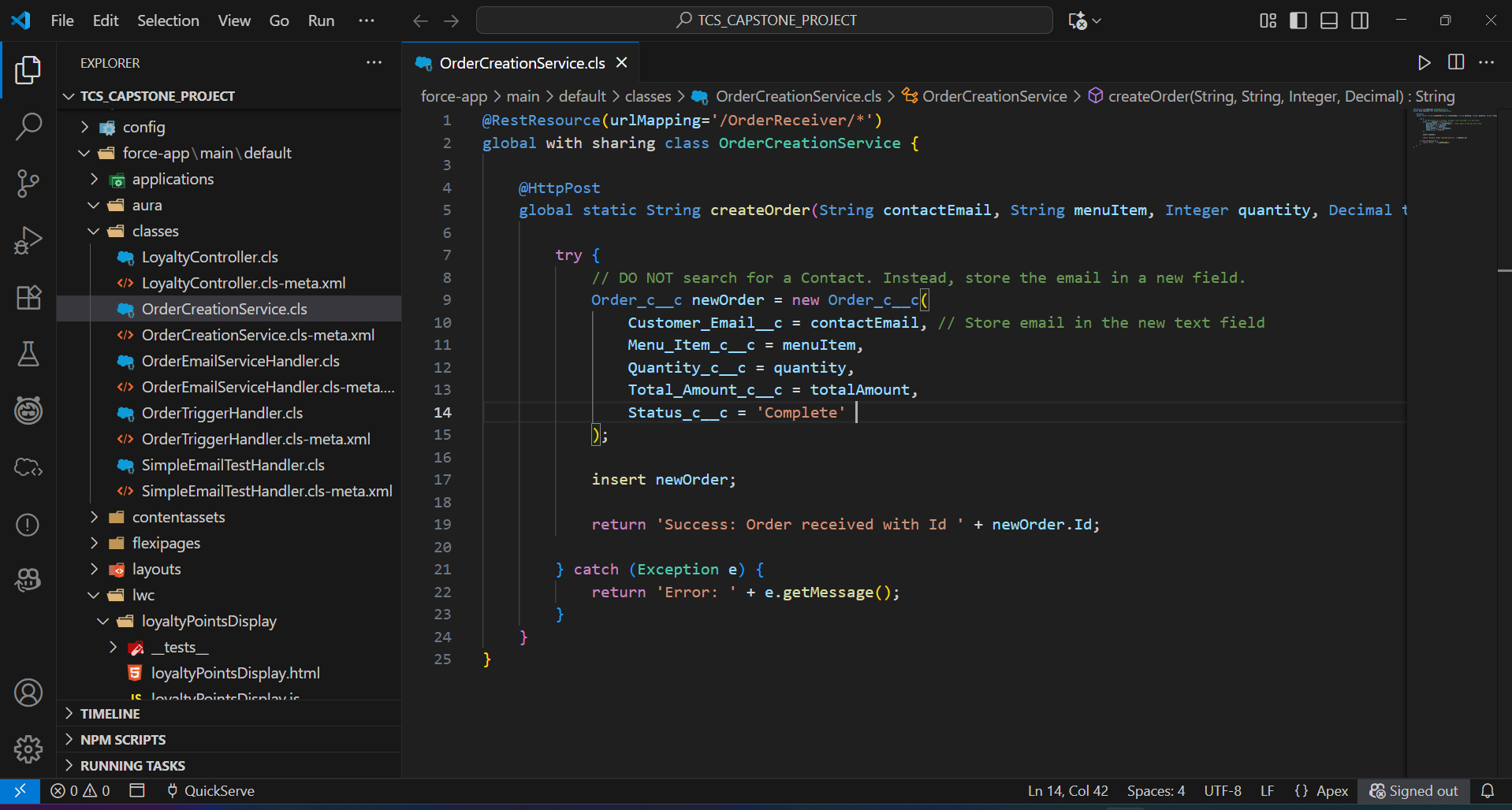
**PHASE 7 - Integration & External Access**

This phase focuses on enabling external systems, like a Point-of-Sale (POS) terminal, to create records in Salesforce. Due to org limitations preventing the use of a standard Connected App, an alternative integration method using a public web service was implemented.

**🔶 Web Services (REST)**

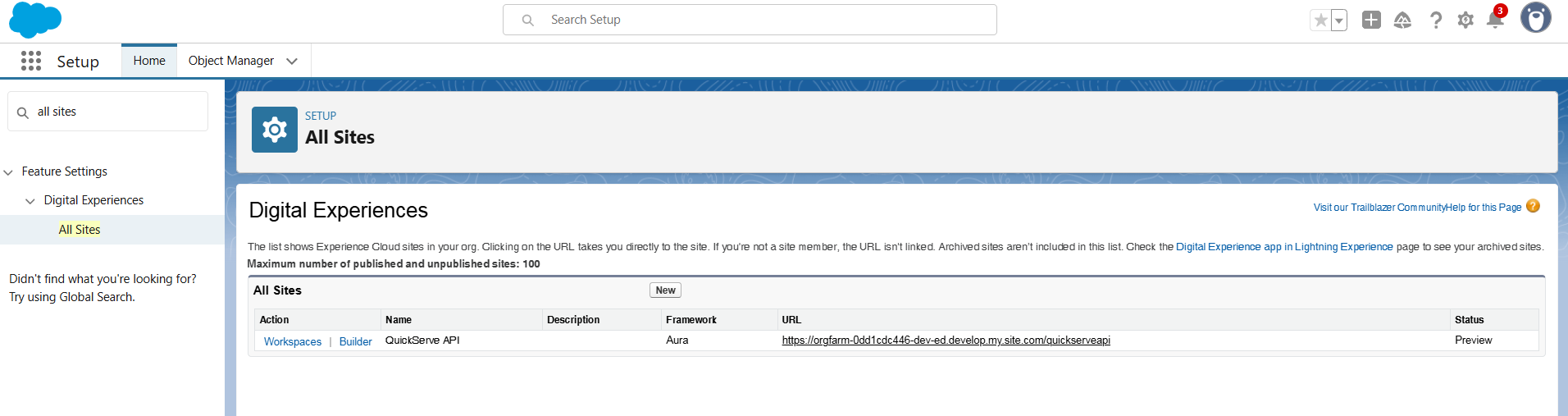
A custom Apex REST service was created to expose an endpoint that external systems can send data to. This allows for an "inbound" integration where a POS system can automatically create an Order\_\_c record in Salesforce.



* **Apex Class**: An @RestResource class named **OrderCreationService** was developed to handle incoming POST requests. The **@RestResource** annotation exposes the class as a web service, and the **@HttpPost** annotation specifies that a method will run when it receives an HTTP POST request.
* **Functionality**: The service is designed to receive a JSON payload containing order details (contact email, menu item, quantity, total amount). It then deserializes this data, creates a new Order\_\_c record, and returns a success or error message.

**🔶 Public Endpoint (Digital Experience)**

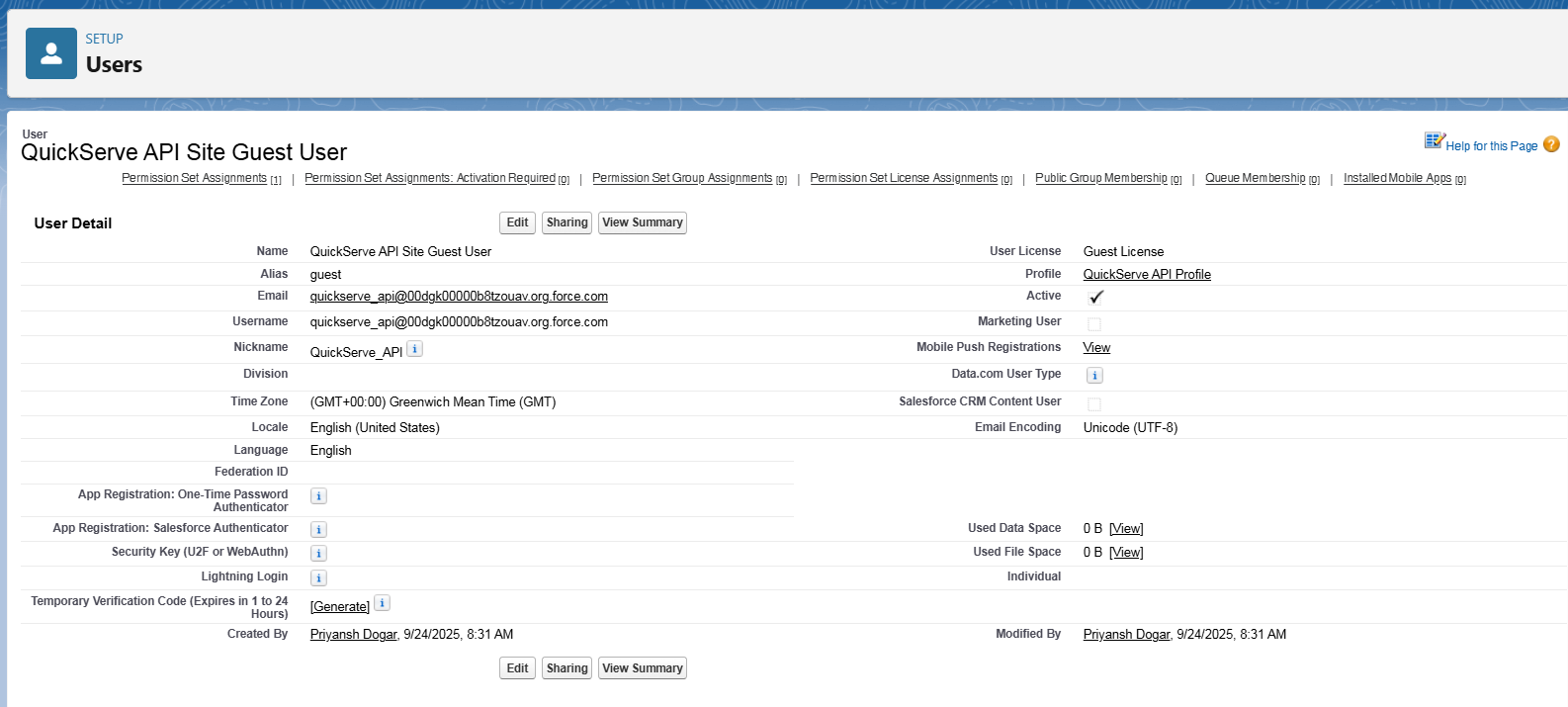
Instead of a private API secured by OAuth, a public endpoint was created using a Salesforce Site (Digital Experience). This provides a publicly accessible URL for the web service.



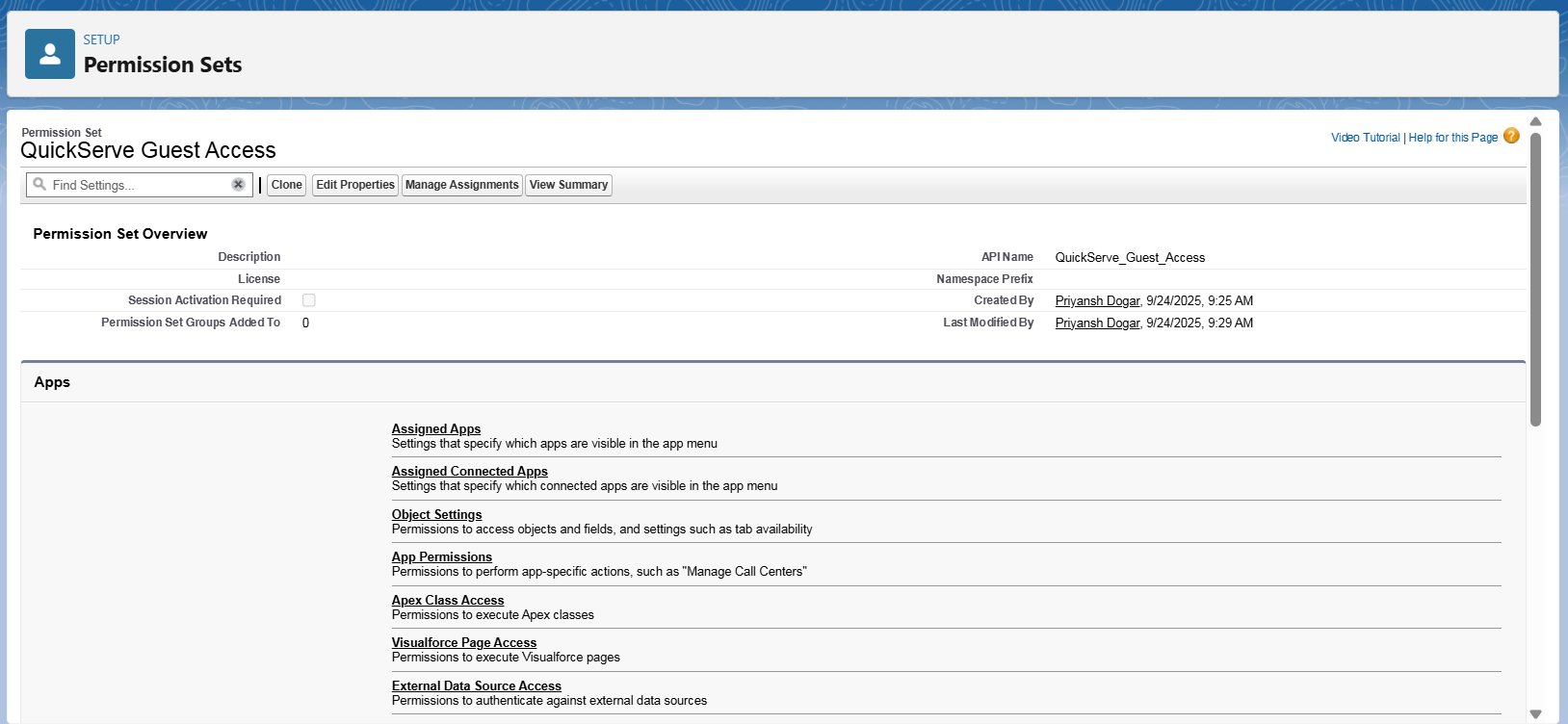
* **Site Setup**: A public Digital Experience site named "QuickServe API" was created to host the web service. This gives the endpoint a stable, public-facing URL.
* **URL Mapping**: The OrderCreationService was exposed via a custom URL mapping (/OrderReceiver/\*) on this public site. This means the final endpoint becomes <https://orgfarm-0dd1cdc446-dev-ed.develop.my.site.com/quickserveapi/services/apexrest/OrderReceiver/>

**🔶 Security & Permissions**

Since a standard Connected App could not be used, security was configured using the Guest User Profile and a dedicated Permission Set. This aligns with Salesforce's enhanced security model for public sites.



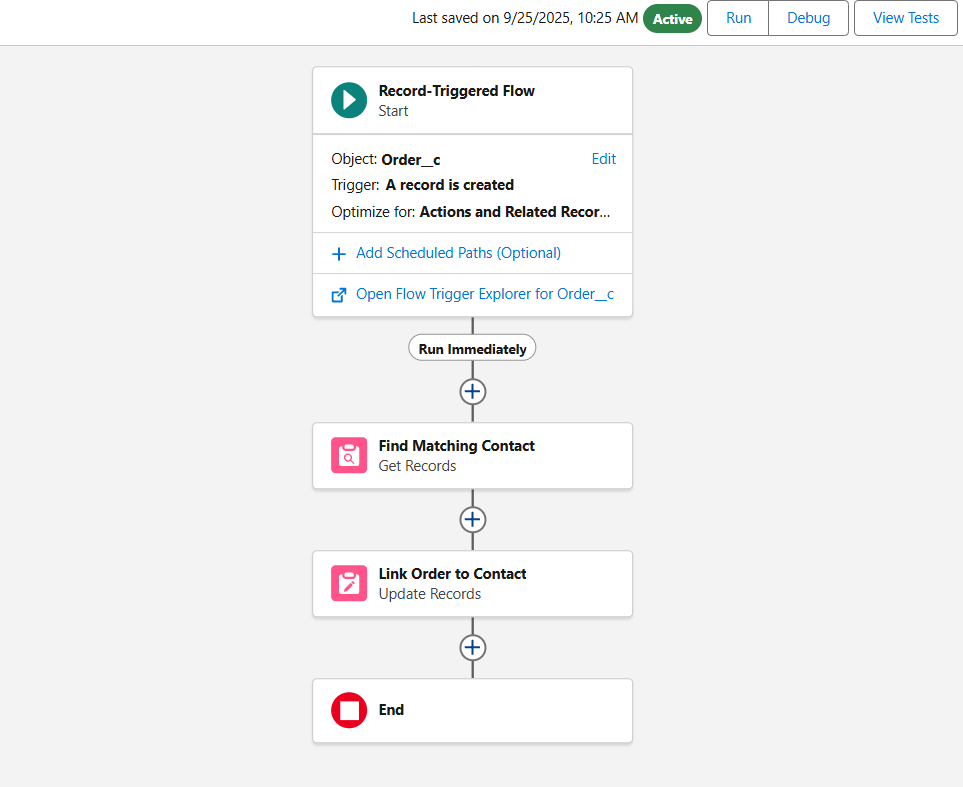
* **Guest User Profile**: The site has a specific "QuickServe API Profile" for unauthenticated users. This profile is intentionally kept with minimal permissions.



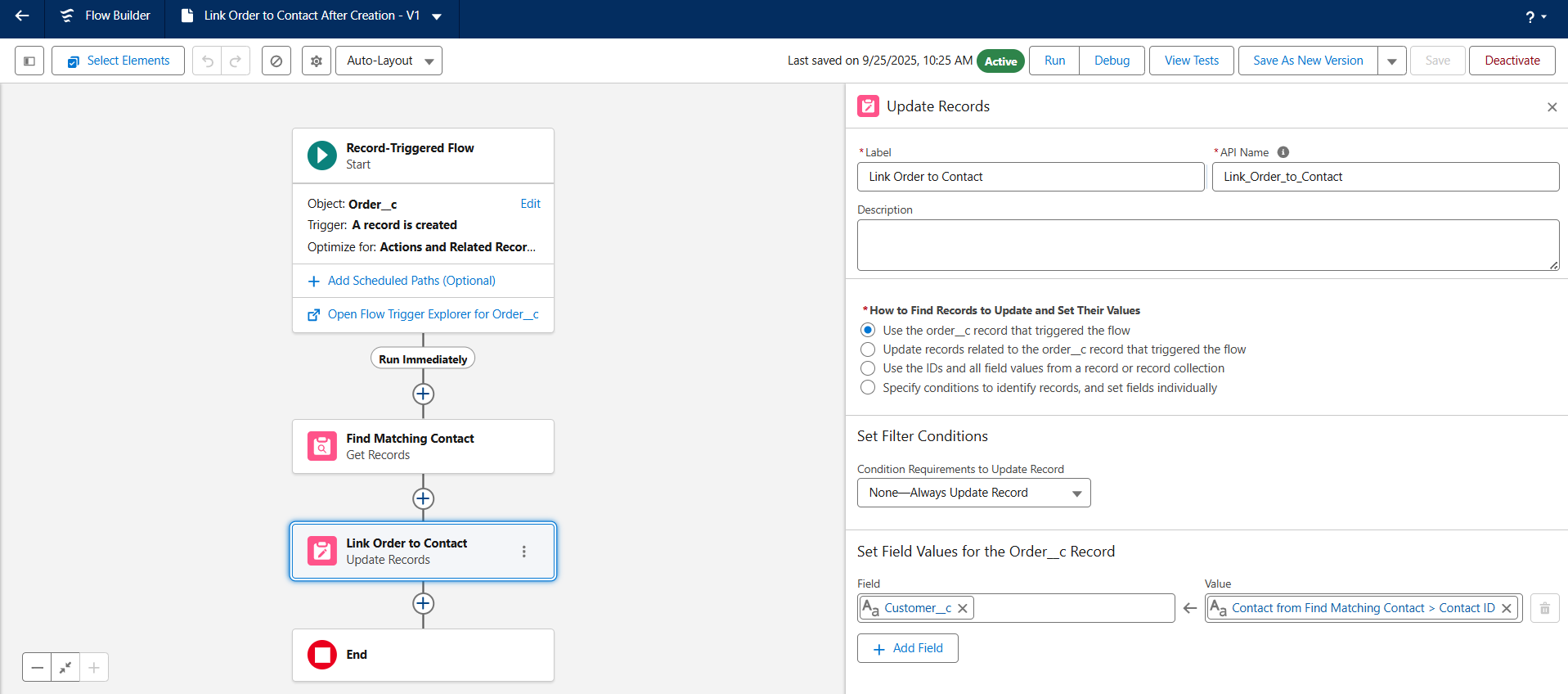
* **Permission Set**: A Permission Set named "QuickServe Guest Access" was created to securely grant the Guest User Profile explicit **Create** access on the Order\_\_c object and **Read** access on the Contact object, along with the necessary field-level security. This is the modern, secure way to grant permissions to guest users, rather than assigning them directly to the profile.

**🔶 Record Linking Automation**

A key challenge is that the secure Guest User cannot query for existing Contact records. A two-step automation process was built to bypass this limitation.



1. The **OrderCreationService** runs in the guest user's context. It creates an Order record but only stores the customer's email in a temporary text field, without linking it to a Contact.
2. An after-save **Record-Triggered Flow** then runs in **System Context**. Because it runs in System Context, it has permission to see all records. It takes the email from the new Order record, queries for the matching Contact, and then populates the Customer\_\_c lookup field, creating the final link.



**🔶 API Limits**

The integration was designed with Salesforce's strict governor and API limits in mind.

* The current Apex service creates one record at a time, which is safe for low-volume transactions. For any future high-volume needs, the service would be enhanced to handle a list of orders in a single transaction (bulkification) to avoid hitting execution limits.