

Maggdalena Larment
Salesforce
Developer

ans



Legarant
DEPLOYER UNE APPLICATION
SALESFORCE



Magdalena Larmet
Salesforce
Developer

DEPLOYER UNE APPLICATION
SALESFORCE



Sommaire

- INTRO** Rappel du projet
- 1** Environnement Salesforce
- 2** Environnement Postman
- 3** Environnement Heroku
- 4** Environnement Postgr  
- 5** D  ploiement et Impl  m  tations
- CONCLU** D  mo et Conclusion





Intro

Rappels du projet



Rappel du projet

22-08-2023

Date démarrage

18-09-2023

Date de fin

À DÉFINIR

Coût du projet

BESOINS EXPRIMÉS

Suite au rachat la société AXG, LEGARANT souhaite conserver son CRM Salesforce, et intégrer les données importantes du CRM AXG à celui-ci.

EXIGENCES DU PROJET

- ✓ Respecter les règles de gestion de la société
- ✓ Effectuer les appels API en respectant les standards Salesforce
- ✓ Utiliser une PAAS parmi les suivantes:
 - Heroku,
 - Azure
 - AWS



Rappel des règles de gestion

Règle de Gestion	Salesforce	@Http	Postman
Création d'un contact : quand un contact est créé dans CRM AXG, il faut vérifier si l'email existe déjà. i. S'il existe, mettre l'id de Salesforce dans le CRM AXG ii. S'il n'existe pas, créer un nouveau contact dans Salesforce et prendre l'id.	createNewContact()	httpPost	CreateContact
Modification d'un contact : quand un champ est modifié sur le Contact dans le CRM AXG, le changement doit être envoyé	updateContact()	httpPatch	UpdateContact
Suppression d'un contact : un contact ne doit pas être supprimé dans Salesforce, il faut le mettre en désactivé	deactivateContact()	httpDelete	DisableContact
Création d'un contrat : quand un contrat est créé dans le CRM AXG, les informations sont envoyées vers Salesforce et l'Id de Salesforce est sauvegardé.	createNewContract()	httpPost	CreateContract
Modification d'un contrat : à chaque modification du contrat, il faut mettre à jour le contrat dans Salesforce en se basant sur l'Id de Salesforce.	updateContract()	httpPatch	UpdateContract

Étapes du projet

STEP 01

ENVIRONNEMENT SALESFORCE

- Création de l'application Connectée
- Création de Champs Personnalisés
- Flow pour gérer External ID
- Création des classes @RestRessource
- Test des méthodes Rest
- Autoriser l'org dans VSC



STEP 02

ENVIRONNEMENT POSTMAN

- Connecter avec l'org Salesforce (Consumer Key, Consumer Secret)
- Création d'une Collection
- Obtention du Token
- Effectuer des requêtes demandées



STEP 03

ENVIRONNEMENT HEROKU

- Création de l'application Heroku
- Connecter avec l'organisation SF
- Déployer sur Github et Connecter le Github créé pour le projet



STEP 04

ENVIRONNEMENT PGADMIN

- Création d'un serveur en renseignant les Crédentials Heroku
- Création de requêtes tests avant le déploiement

Étapes du projet



APPELS API
CRM ADX

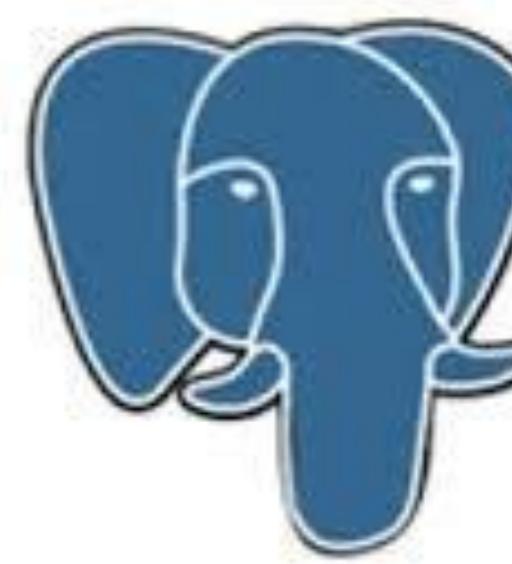


App connectée Salesforce

SYNC MOBILE APP



TEST SUR BDD TEST





Environnement Salesforce

Projet mené pour la

Fasha

FASHA

Projet mené pour la Société LEGARANT



Environnement Postman

Connecter Salesforce et Postman
Consumer Key & Consumer Salesforce
LegarantConnect

The screenshot shows the Salesforce Setup interface under the 'Manage Connected Apps' section. A specific connected app named 'LegarantConnect' is selected. The app's details are listed, including its version (1.0), API name (LegarantConnect), and creation date (24/08/2023 06:19). The consumer key and secret are also displayed. The OAuth settings are configured for Full access (full) with a callback URL of https://oauth.pstmn.io/v1/callback.

Consumer Details

Consumer Key: 3MVG98Gq08Po4ZmxXGJdBR2N93PL3m4thjx4alPRRodaZX_gZU6vDJ_m_K3bt68UIBSG_aEAhfchhvjTx4Lg
Consumer Secret: E5DFE9A5F2A61CE1369526D772859B04D23C3483D43B600DB6E86D6B7FF011B0

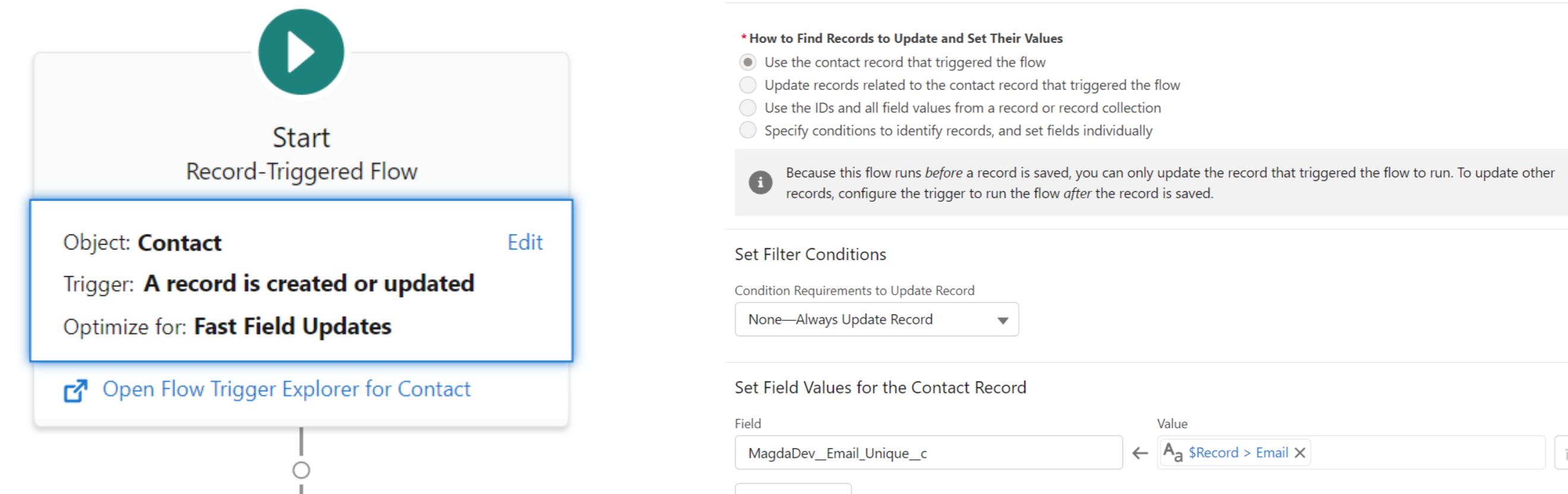
Staged Consumer Details

Generate Apply Cancel



Environnement Salesforce

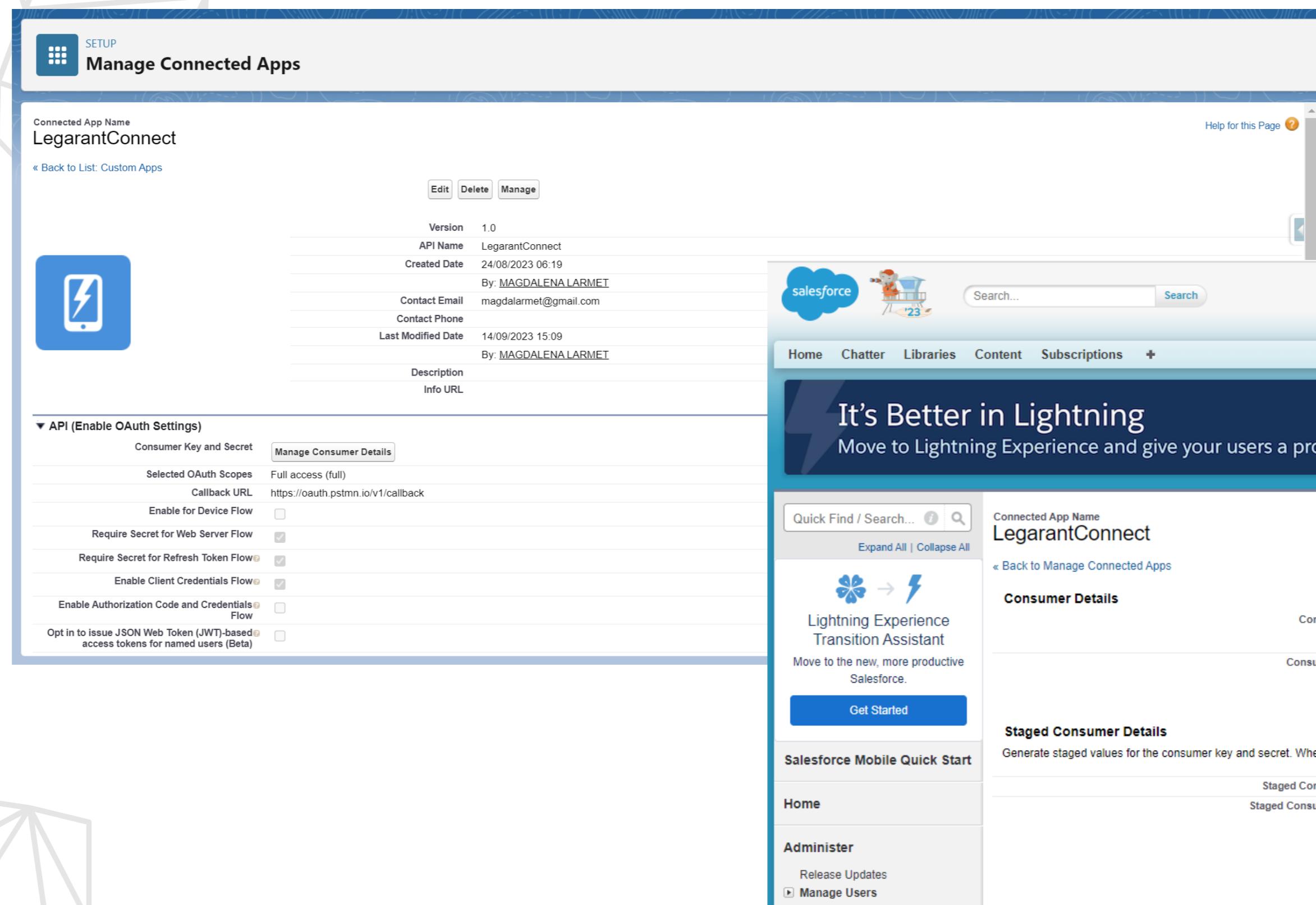
Flow Builder
Gestion de l'email Unique
À la Création et à l'Update



Le champ email se remplit automatiquement avec l'email du contact qui correspond à l'external ID

Environnement Salesforce

App Manager
Connected App
LegarantConnect



The screenshot shows the 'Manage Connected Apps' page for the 'LegarantConnect' app. The app details are as follows:

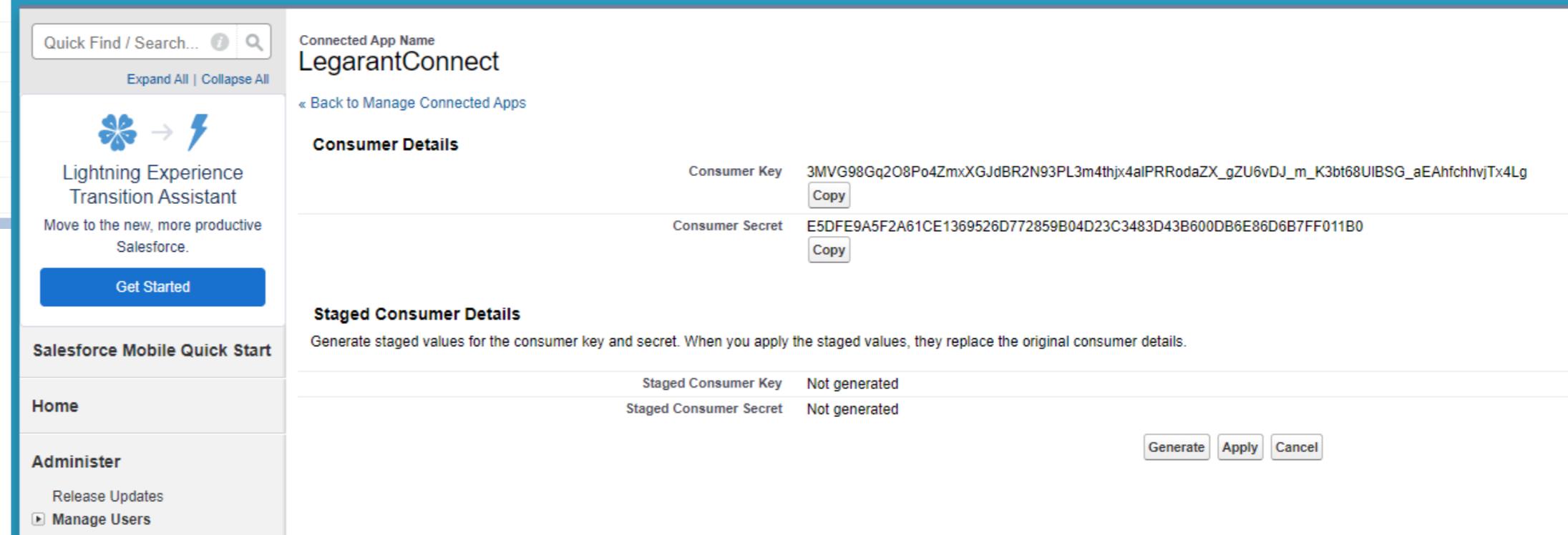
- Connected App Name: LegarantConnect
- Version: 1.0
- API Name: LegarantConnect
- Created Date: 24/08/2023 06:19
- Contact Email: MAGDALENA.LARMET@gmail.com
- Contact Phone:
- Last Modified Date: 14/09/2023 15:09
- Description: Info URL

The 'API (Enable OAuth Settings)' section is expanded, showing the following configuration:

- Consumer Key and Secret: Manage Consumer Details
- Selected OAuth Scopes: Full access (full)
- Callback URL: https://oauth.pstmn.io/v1/callback
- Enable for Device Flow:
- Require Secret for Web Server Flow:
- Require Secret for Refresh Token Flow:
- Enable Client Credentials Flow:
- Enable Authorization Code and Credentials Flow:
- Opt in to issue JSON Web Token (JWT)-based access tokens for named users (Beta):

On the right side of the interface, there is a promotional banner for Lightning Experience and a sidebar with links like Home, Chatter, Libraries, Content, Subscriptions, and a 'Lightning Experience Transition Assistant' section.

It's Better in Lightning
Move to Lightning Experience and give your users a productivity boost.



The screenshot shows the 'Consumer Details' section of the 'LegarantConnect' connected app configuration. It displays the consumer key and consumer secret values.

Consumer Details

Consumer Key	3MVG98Gq2O8Po4ZmxXGJdBR2N93PL3m4thjx4alPRRodaZX_gZU6vDJ_m_K3bt68UIBSG_aEAhfchhvjTx4Lg
Consumer Secret	E5DfE9A5F2A61CE1369526D772859B04D23C3483D43B600DB6E86D6B7FF011B0

Staged Consumer Details
Generate staged values for the consumer key and secret. When you apply the staged values, they replace the original consumer details.

Staged Consumer Key	Not generated
Staged Consumer Secret	Not generated

Buttons: Generate, Apply, Cancel



Environnement Salesforce

Le code @RestRessources ContactService



```
@RestResource(urlMapping='/Contact/*')
global with sharing class ContactService {

    @HttpGet
    global static Contact getContactById() {
        RestRequest request = RestContext.request;
        // Get the contactId from the end of the URL
        String contactId = request.requestURI.substring(
            request.requestURI.lastIndexOf('/') + 1);
        //Id contactId = RestContext.request.params.get('Id');

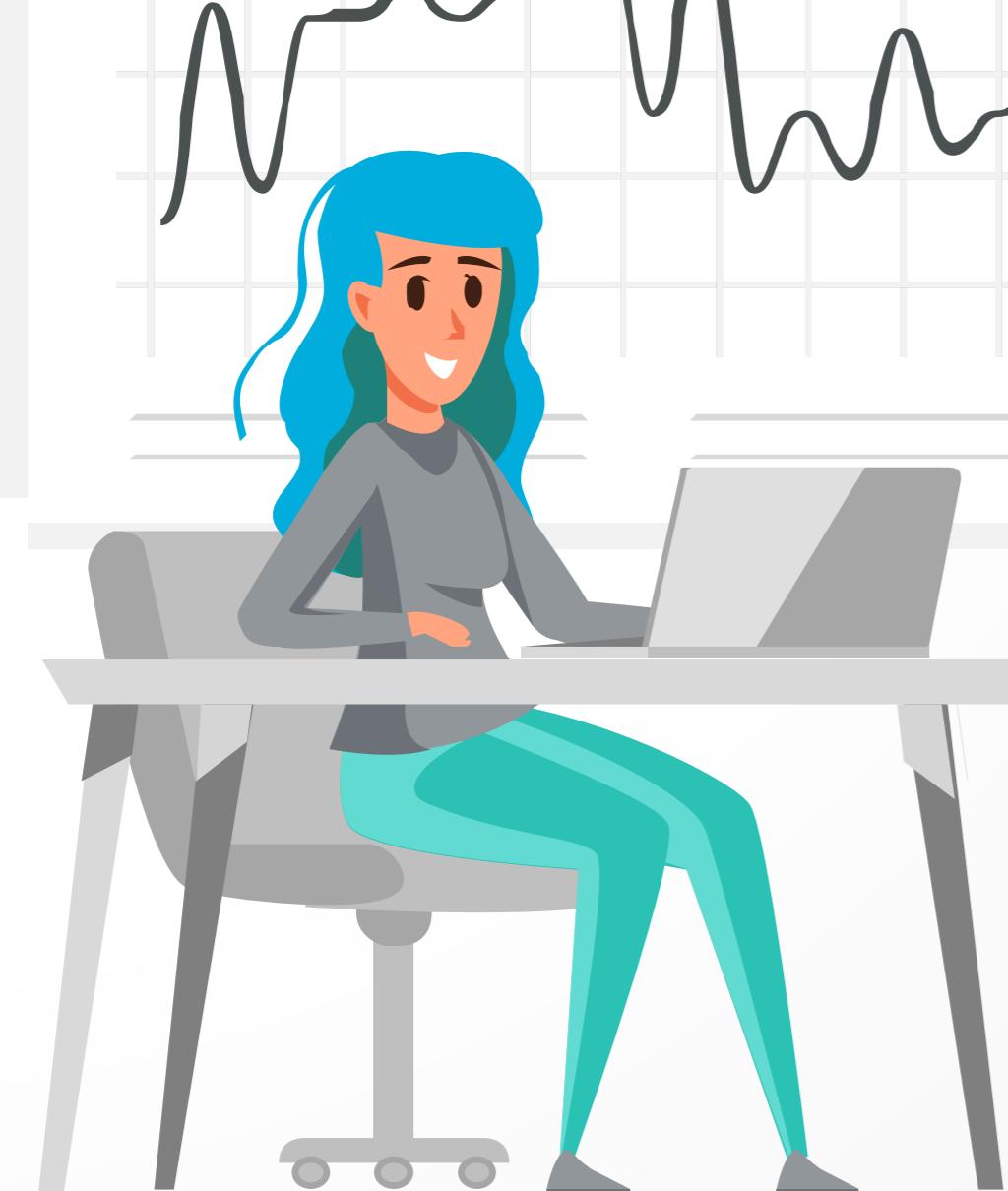
        Contact result = [
            SELECT Salutation, FirstName, LastName, Email, Phone
            FROM Contact
            WHERE Id = :contactId
        ];
        return result;
    }

    @HttpPost
    global static String createNewContact() {
        RestRequest request = RestContext.request;
        Map<String, Object> requestResult = (Map<String, Object>) JSON.deserializeUntyped(request.requestBody.toString());
        List<Object> contacts = (List<Object>) requestResult.get('records');
        List<Contact> contactsToInsert = new List<Contact>();
        for (Object contactObject : contacts) {
            Map<String, Object> parsedContact = (Map<String, Object>) contactObject;
            Contact thisContact = new Contact(
                Active__c = true,
                FirstName = (String) parsedContact.get('FirstName'),
                LastName = (String) parsedContact.get('LastName'),
                Email = (String) parsedContact.get('Email'),
                Email_Unique__c = (String) parsedContact.get('Email')
            );
            contactsToInsert.add(thisContact);
        }
        Database.upsert(contactsToInsert, Contact.Email_Unique__c, true);

        Map<String, String> stringMap = new Map<String, String>();
        for (Contact contact : contactsToInsert) {
            if (contact.Id != null) {
                stringMap.put(contact.Id, contact.Email_Unique__c);
            }
        }
        return JSON.serializePretty(stringMap);
    }
}
```

Environnement Salesforce

Le code @RestRessources
ContactService



```
@HttpPatch
global static Id updateContact() {

    // Add the Contact Id as a parameter in the URL
    RestRequest request = RestContext.request;
    String contactId = request.requestURI.substring(request.requestURI.lastIndexOf('/') + 1);

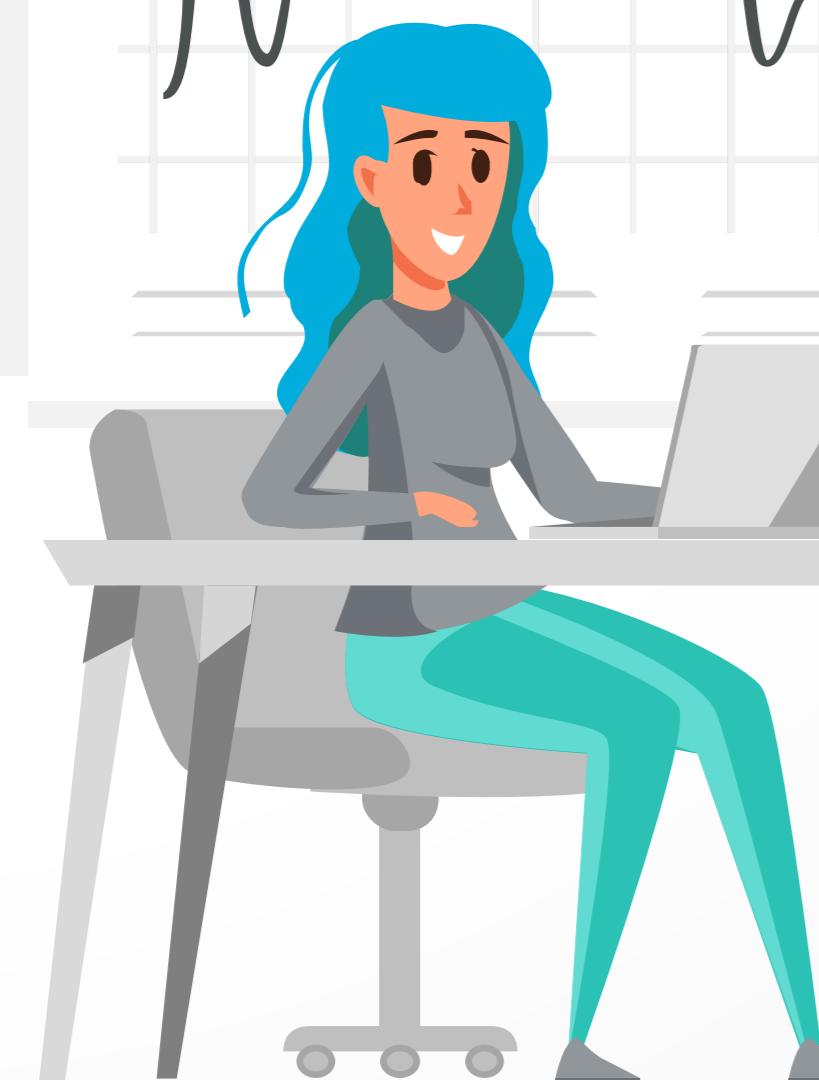
    // Check if contact exist
    List<Contact> matchingContacts = [SELECT Id FROM Contact WHERE Id = :contactId];
    if (matchingContacts.isEmpty()) {
        // Case : ID does not exist
        // return an error
        return 'Contact with ID ' + contactId + ' does not exist';
    }

    Contact thisContact = matchingContacts[0];
    //convert JSON data from request
    Map<String, Object> params = (Map<String, Object>) JSON.deserializeUntyped(request.requestBody.toString());
    //update fields on object
    for (String fieldName : params.keySet()) {
        thisContact.put(fieldName, params.get(fieldName));
    }

    update thisContact;
    return thisContact.Id;
}
```

Environnement Salesforce

Le code @RestRessources
ContactServiceTest



```
@isTest
public class ContactServiceTest {
    private static String contactFirstName = 'Magda';
    private static String contactLastName = 'Test';

    @testSetup
    static void setupForTest(){

    }

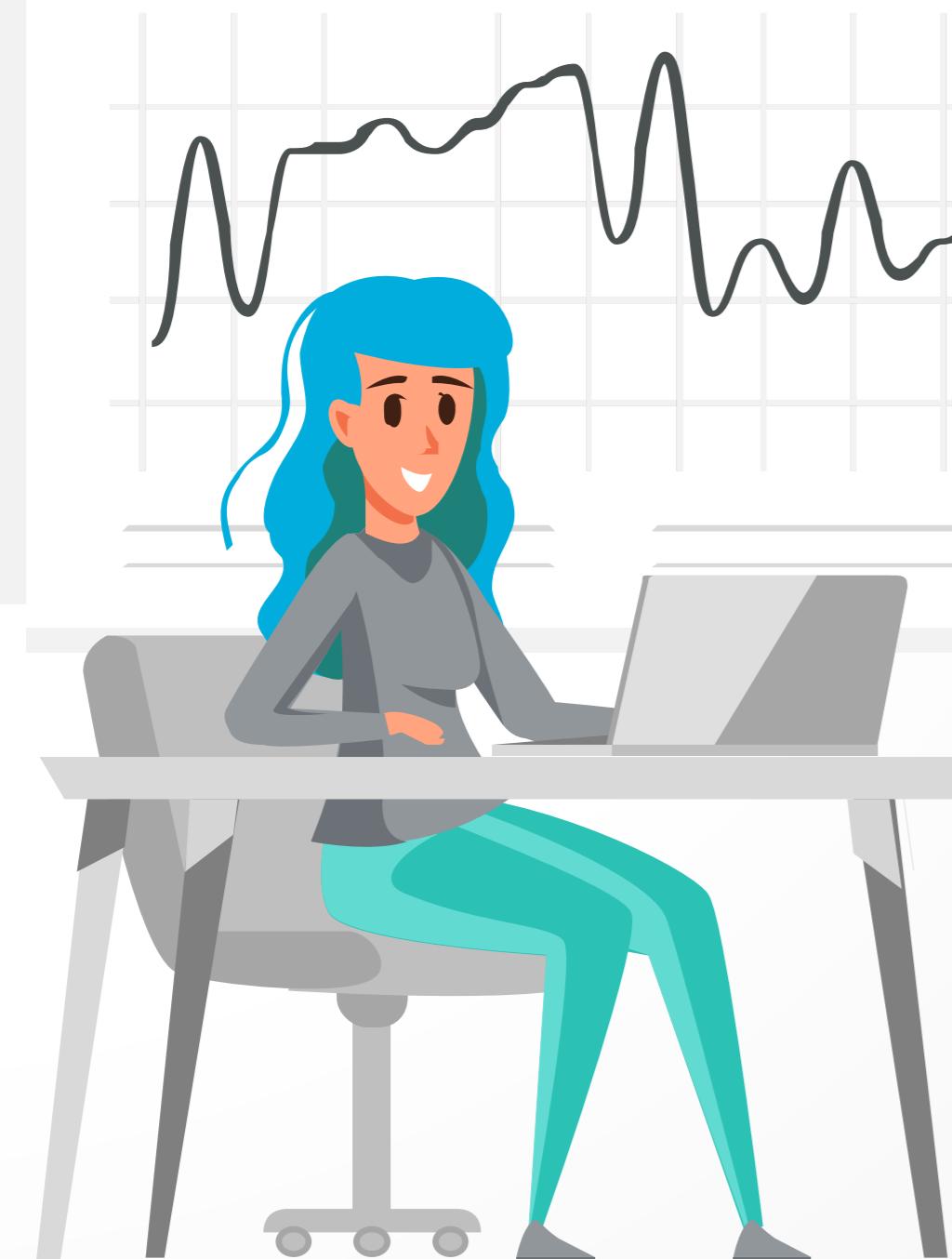
    @isTest static void testGetContactById() {
        List<Contact> contactListForThisTest = TestDataFactory.createContacts(2);
        Id recordId = contactListForThisTest.get(0).Id;
        Test.startTest();
        // Set up a test request
        RestRequest request = new RestRequest();
        request.requestUri ='https://legarant4-dev-ed.lightning.force.com/services/apexrest/Contact/' + recordId;
        request.httpMethod = 'GET';
        RestContext.request = request;

        // Call the method to test
        Contact thisContact = ContactService.getContactById();
        Test.stopTest();

        // Verify results
        System.assert(thisContact != null);
    }
}
```

Environnement Salesforce

Le code @RestRessources
ContractService



```
@HttpPost
global static String createNewContract() {
    RestRequest request = RestContext.request;
    Map<String, Object> requestResult = (Map<String, Object>) JSON.deserializeUntyped(request.requestBody.toString());
    List<Object> contracts = (List<Object>) requestResult.get('records');
    List<Contract> contractsToInsert = new List<Contract>();

    for (Object contractObject : contracts) {
        Map<String, Object> parsedContract = (Map<String, Object>) contractObject;
        Contract thisContract = new Contract(
            Name = (String) parsedContract.get('Name'),
            Status = (String) parsedContract.get('Status'),
            AccountId = Id.valueOf((String) parsedContract.get('AccountId')),
            StartDate = Date.valueOf((String) parsedContract.get('StartDate')),
            ContractTerm = Integer.valueOf((String) parsedContract.get('ContractTerm'))
        );
        contractsToInsert.add(thisContract);
    }
    Database.upsert(contractsToInsert, Contract.Id, true);

    Map<String, Object> stringMap = new Map<String, Object>();
    for (Contract contract : contractsToInsert) {
        if (contract.Id != null) {
            stringMap.put(contract.Id, contract.Name);
        }
    }
    return JSON.serializePretty(stringMap);
}
```

Environnement Salesforce

Le code @RestRessources ContractService

```
@HttpPatch
global static Id updateContract(){
    // extract the Contract Id as a parameter in the URL
    RestRequest request = RestContext.request;
    String contractId = request.requestURI.substring(request.requestURI.lastIndexOf('/')+1);

    // Check if contact exist
    List<Contract> matchingContracts = [SELECT Id, AccountId FROM Contract WHERE Id = :contractId];
    if (matchingContracts.isEmpty()) {
        // Case : ID does not exist
        // return an error
        return 'Contract with ID ' + contractId + ' does not exist';
    }

    Contract thisContract = matchingContracts[0];
    //convert JSON data from request
    Map<String, Object> requestResult = (Map<String, Object>) JSON.deserializeUntyped(request.requestBody.toString());

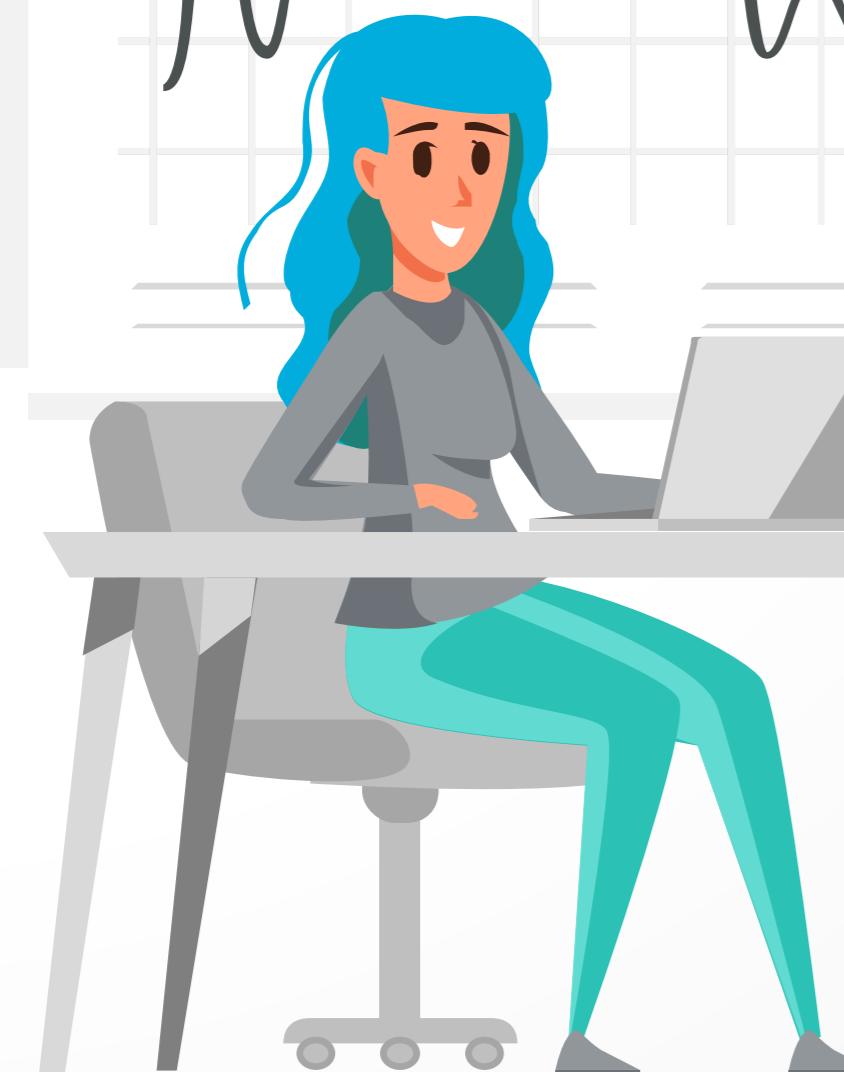
    //update fields on object
    for (String fieldName : requestResult.keySet()) {
        thisContract.put(fieldName, requestResult.get(fieldName));
    }

    update thisContract;
    return thisContract.Id;
}
```



Environnement Salesforce

Le code @RestRessources
ContractServiceTest



```
@isTest

public class ContractServiceTest {
    @testSetup
    static void setupForTest(){

    }

    @isTest static void testGetContractById() {
        Id contractForThisTest = TestDataFactory.createAccountAndContractRecord();
        Id recordId = contractForThisTest;
        Test.startTest();
        // Set up a test request
        RestRequest request = new RestRequest();
        request.requestUri ='https://legarant-b-dev-ed.lightning.force.com/services/apexrest/Contact/' + recordId;
        request.httpMethod = 'GET';
        RestContext.request = request;

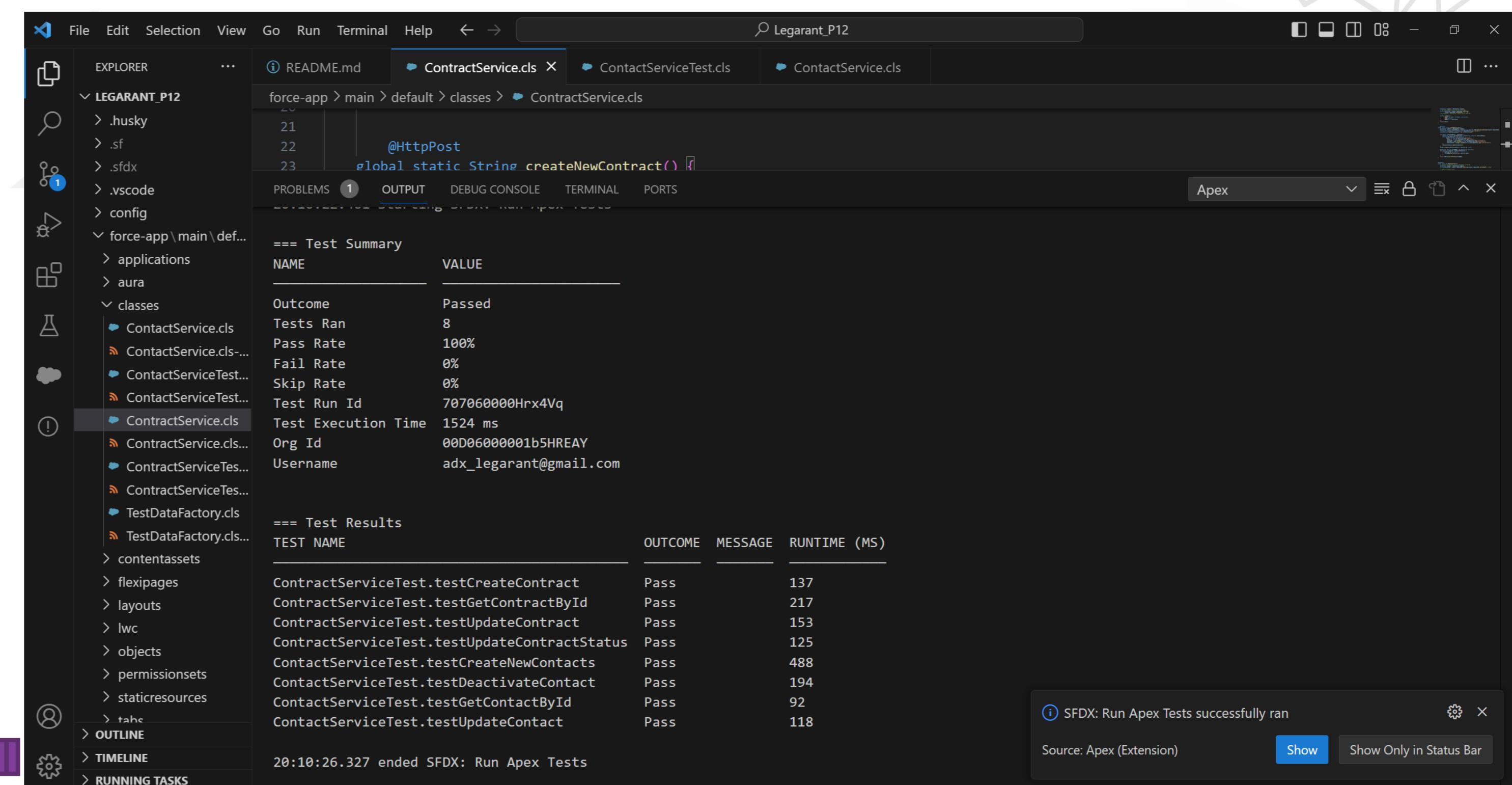
        // Call the method to test
        Contract thisContract = ContractService.getContractById();
        Test.stopTest();

        // Verify results
        System.assert(thisContract != null);
    }

    @isTest
    static void testCreateContract() {
```

Environnement Salesforce

Le code @RestRessources
Résultat des Test passés



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "LEGARRANT_P12".
- Editor:** Displays the Apex class `ContractService.cls` with a method annotated with `@HttpPost` and `global static String createNewContract()`.
- Output Panel:** Shows the "Apex" tab with the "Test Results" section.
- Test Results:**

TEST NAME	OUTCOME	MESSAGE	RUNTIME (MS)
ContractServiceTest.testCreateContract	Pass		137
ContractServiceTest.testGetContractById	Pass		217
ContractServiceTest.testUpdateContract	Pass		153
ContractServiceTest.testUpdateContractStatus	Pass		125
ContactServiceTest.testCreateNewContracts	Pass		488
ContactServiceTest.testDeactivateContact	Pass		194
ContactServiceTest.testGetContactById	Pass		92
ContactServiceTest.testUpdateContact	Pass		118
- Status Bar:** Shows the message "SFDX: Run Apex Tests successfully ran".

Environnement Salesforce

Le code @RestRessources

Couverture du code

Overall Code Coverage			
Class	Percent	Lines	
Overall	93%		
ContactService	97%	46/47	
ContractService	97%	38/39	
TestDataFactory	83%	25/30	



The screenshot shows the Visual Studio Code (VS Code) interface with the Apex extension installed. The workspace is set up for a Salesforce project named 'LEGANT_P12'. The Explorer sidebar on the left lists various project files and components. The main editor area displays an Apex class named 'ContractService.cls' with a single method annotated with '@HttpPost'. Below the code, the 'PROBLEMS' tab shows one error. The 'OUTPUT' tab displays the results of an Apex test run.

PROBLEMS 1

OUTPUT

DEBUG CONSOLE TERMINAL PORTS

Apex

==== Test Summary

NAME	VALUE
Outcome	Passed
Tests Ran	8
Pass Rate	100%
Fail Rate	0%
Skip Rate	0%
Test Run Id	707060000Hrx4Vq
Test Execution Time	1524 ms
Org Id	00D06000001b5HREAY
Username	adx_legarant@gmail.com

==== Test Results

TEST NAME	OUTCOME	MESSAGE	RUNTIME (MS)
ContractServiceTest.testCreateContract	Pass		137
ContractServiceTest.testGetContractById	Pass		217
ContractServiceTest.testUpdateContract	Pass		153
ContractServiceTest.testUpdateContractStatus	Pass		125
ContactServiceTest.testCreateNewContacts	Pass		488
ContactServiceTest.testDeactivateContact	Pass		194
ContactServiceTest.testGetContactById	Pass		92
ContactServiceTest.testUpdateContact	Pass		118

20:10:26.327 ended SFDX: Run Apex Tests

SFDX: Run Apex Tests successfully ran

Source: Apex (Extension)

Show

Show Only in Status Bar





2



Projet mené pour la Société Fasha



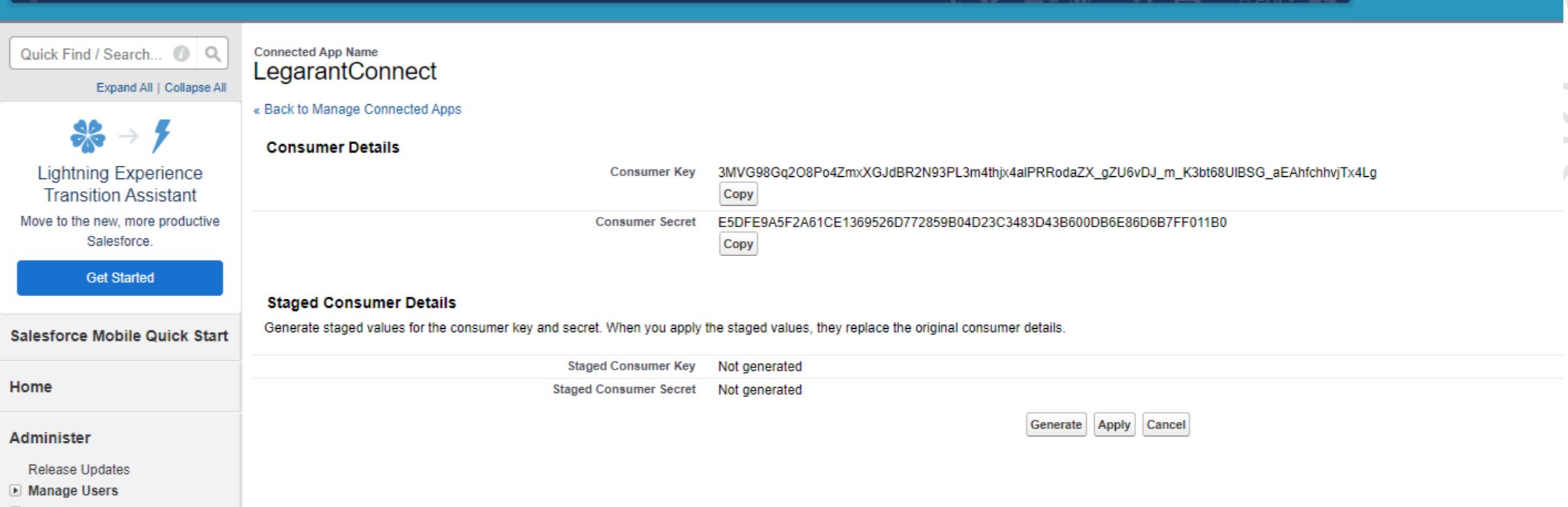
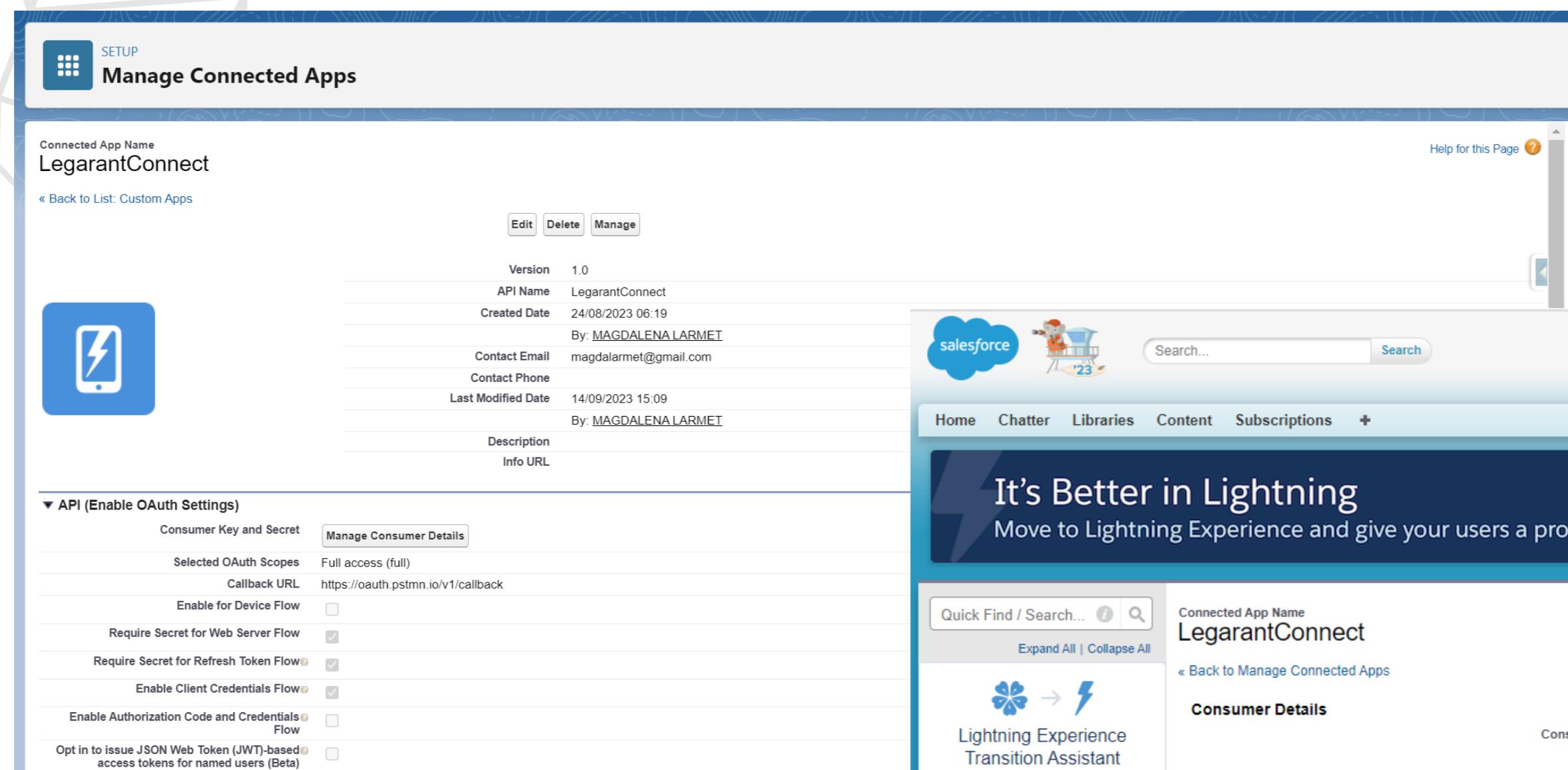
Environnement Postman

Projet mené pour la Société LEGARANT



Environnement Postman

Appels d'API
Identifiants de connexion
LegarantConnect



Environnement Postman

Application Salesforce Création des variables

The screenshot shows the Postman interface with the 'Variables' tab selected in the 'LegarantCollection' workspace. The table displays the following environment variables:

	Variable	Initial value	Current value
url	https://login.salesforce.com	https://login.salesforce.com	
version	58.0	58.0	
username			
password			
secretToken			
clientId	3MVG9_kZcLde7U5oE.rw...	3MVG98_Psg5cppyYCmk1gZNC25o0OSXpgpodIS29IZ6pXiHkt...	
clientSecret	96DB79DB60E20092FC53...	96DB79DB60E20092FC53DDCEC07F2F06C8D2BF961A75F...	
redirectUrl	https://legarant-b-dev-ed.....	https://legarant-b-dev-ed.develop.my.salesforce.com	
initAccessToken			
site			
_accessToken			
_endpoint		https://legarant-b-dev-ed.develop.my.salesforce.com	

Environnement Postman

La collection Legarant
Création de contacts

The screenshot shows the Postman application interface. The left sidebar displays 'My Workspace' with a collection named 'LegarantCollection'. The main area shows a POST request titled 'CreateContacts' under the 'LegarantCollection' folder. The request URL is `{_endpoint}/services/apexrest/Contact/`. The 'Body' tab is selected, showing a JSON payload with three records:

```
1  {
2    "records": [
3      {
4        "attributes": {"type": "Contact"},
5        "FirstName": "Name-30",
6        "LastName": "LastName-30",
7        "Email": "Name-30@gmail.com"
8      },
9      {
10        "attributes": {"type": "Contact"},
11        "FirstName": "Name-31",
12        "LastName": "LastName-31",
13        "Email": "Name-31@gmail.com"
14      },
15      {
16        "attributes": {"type": "Contact"},
17        "FirstName": "Name-32",
18        "LastName": "LastName-32",
19        "Email": "Name-32@gmail.com"
20      }
21  ]
```

The response status is 200 OK with a time of 354 ms and a size of 553 B. The response body contains three email addresses: Name-30@gmail.com, Name-31@gmail.com, and Name-32@gmail.com.



Environnement Postman

La collection Legarant
Mise à jour de contacts

The screenshot shows the Postman interface with a PATCH request. The URL is `{_endpoint}/services/apexrest/{SOBJECT_API_NAME}/{RECORD_ID}`. The Body tab contains the JSON payload: `{"LastName": "ChangeLastName"}`. Path Variables are set to SOBJECT_API_NAME: Contact and RECORD_ID: 0030600002BQIWZAA5.

The screenshot shows the Postman interface after executing the PATCH request. The status is 200 OK, and the response body is `"0030600002BQIWZAA5"`.



Environnement Postman

La collection Legarant
Suppression de contacts

HTTP LegarantCollection / Delete/Deactivate Contact

DELETE `{$_endpoint}/services/apexrest/:SOBJECT_API_NAME/:RECORD_ID`

Params • Authorization Headers (12) Body • Pre-request Script Tests Settings

Query Params

Key	Value
Key	Value

Path Variables

Key	Value
SOBJECT_API_NAME	Contact
RECORD_ID	0030600002BQIWZAA5

HTTP LegarantCollection / Delete/Deactivate Contact

DELETE `{$_endpoint}/services/apexrest/:SOBJECT_API_NAME/:RECORD_ID`

Params • Authorization Headers (12) Body • Pre-request Script Tests Settings

Body

```
1 "Active__c": false
```

Status: 200 OK Time: 318 ms Size: 368 B Save as Example

Body Cookies (3) Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 "0030600002BQIWZAA5"
```

Postbot Runner Start Proxy Cookies Trash



Environnement Postman

La collection Legarant
Création de contrats

The screenshot shows the Postman application interface. On the left, the sidebar displays 'My Workspace' with a collection named 'LegrantCollection'. Under this collection, several requests are listed: 'POST CreateContacts', 'POST CreateContact', 'POST CreateContracts' (which is selected), 'PATCH ActivateContract', 'PATCH UpdateContractFields', 'PATCH ActivateContact', and 'DEL Delete/Deactivate Contact'. The main panel shows a POST request for 'CreateContracts' with the URL `POST {{_endpoint}}/services/apexrest/Contract/`. The 'Body' tab is selected, showing a JSON payload:

```
1  {
2    "records" : [
3      {
4        "attributes" : {"type" :"Contract"},
5        "Name": "ContractName-26",
6        "Status": "Draft",
7        "AccountId": "0010600002Cw3ZAAJ",
8        "StartDate": "2023-11-01",
9        "ContractTerm": "5"
10      },
11      {
12        "attributes" : {"type" :"Contract"},
13        "Name": "ContractName-27",
14        "Status": "Draft",
15        "AccountId": "0010600002CunKXAAZ",
16        "StartDate": "2024-07-01",
17        "ContractTerm": "6"
18      }
]
```

Below the body, the 'Test Results' tab shows a successful response with status 200 OK, time 254 ms, and size 498 B. The response body is:

```
1  "\n  \"8000600000D29d8AAB\" : \"ContractName-27\", \n  \"8000600000D29d7AAB\" : \"ContractName-26\"\n"
```



Environnement Postman

La collection Legarant
Mise à jour de contrats

The screenshot shows the Postman interface with a PATCH request. The URL is `{_endpoint}/services/apexrest/:SOBJECT_API_NAME/:RECORD_ID`. The Body tab is selected, showing the following JSON:

```
1 "ContractTerm": 8
```

The screenshot shows the Postman interface with a successful response (Status: 200 OK). The Body tab displays the response body: `"8000600000D28niAAB"`.



Environnement Heroku

Environnement Heroku

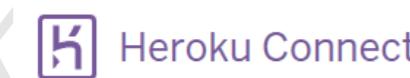
Paramétrage de l'outil
Création de l'application & ajout des add-ons

The screenshot shows the Heroku Platform interface. At the top, there's a header with the Salesforce Platform logo, the Heroku logo, and a "Personal" dropdown. Below the header is a search bar labeled "Filter apps and pipelines". A purple hexagonal icon next to the text "adxlegarant" indicates an attached application. On the right side, there's a "Find more add-ons" button. The main area is titled "Add-ons" and features a search bar "Quickly add add-ons from Elements". Three add-ons are listed: "Heroku Connect" (Attached as DATABASE), "Heroku Postgres" (Attached as LOGTAIL), and "Logtail". To the right of these, there's a summary table with columns for "Demo Edition", "Free", "Mini", "~\$0.007/hour", "Free (5GB/m, 8 days)", "Free", and "\$12". At the bottom left, there's a section for "Estimated Monthly Cost".

Demo Edition	Free	Mini	~\$0.007/hour	Free (5GB/m, 8 days)	Free	\$12

Environnement Heroku

Heroku Connect Mapping des champs



Activity

Past Hour

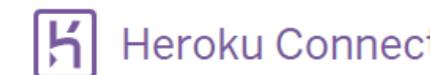
No data available for the selected time period

Mapped Fields

Field Name	Label	Type	Length	Indexed
AccountId	Account ID	reference to: Account	18	
Active_c	Active	boolean	0	
CreatedDate	Created Date	datetime	0	
Email	Email	email	80	Yes
Email_Unique_c	Email Unique	email	80	Yes
FirstName	First Name	string	40	
Id	Contact ID	id	18	Yes
IsDeleted	Deleted	boolean	0	
LastName	Last Name	string	80	
MasterRecordId	Master Record ID	reference to: Contact	18	
Name	Full Name	string	121	
OwnerId	Owner ID	reference to: User	18	
ReportsToId	Reports To ID	reference to: Contact	18	
SystemModstamp	System Modstamp	datetime	0	Yes

Environnement Heroku

Heroku Connect Mapping des champs Contact



MAPPED FIELDS FREQUENCY LAST CHECKED SALESFORCE
11 Polls every 10 minutes 2023-09-22 18:27:08 UTC

Activity

Past Hour ▾

No data available for the selected time period

Mapped Fields

Field Name	Label	Type	Length	Indexed
AccountId	Account ID	reference to: Account	18	
ActivatedDate	Activated Date	datetime	0	
ContractNumber	Contract Number	string	30	
ContractTerm	Contract Term	int	0	
CreatedDate	Created Date	datetime	0	
EndDate	Contract End Date	date	0	
Id	Contract ID	id	18	Yes
IsDeleted	Deleted	boolean	0	
StartDate	Contract Start Date	date	0	
Status	Status	picklist	255	
SystemModstamp	System Modstamp	datetime	0	Yes



Environnement Salesforce

Environnement PGAdmin

Requêtes SQL
Ajout d'un contact

The screenshot shows the pgAdmin 4 interface. The left sidebar is the Object Explorer, displaying various database objects like Domains, FTS Configurations, and Tables (6). The main pane shows a query window with the following SQL code:

```
1 insert into adxlegarant.contact (lastname, firstname, email)
2 VALUES ('Examinateur', 'Openclassrooms', 'examoc@gmail.com');
```

The Messages tab shows the result of the query: "INSERT 0 1". Below it, a message states: "Query returned successfully in 143 msec." At the bottom, it says "Total rows: 35 of 35 Query complete 00:00:00.143". A note at the top right indicates that the user is running version 7.6 of pgAdmin 4, while the current version is 7.7.



Environnement PGAdmin

Requêtes SQL
Ajout d'un contrat

The screenshot shows the pgAdmin 4 interface. The left sidebar is the Object Explorer, displaying various database objects like Domains, FTS Configurations, FTS Dictionaries, etc., and a list of Tables (6) including _hcmeta, _sf_event_log, _trigger_log, _trigger_log_archive, contact, and contract. The contract table is expanded, showing 14 columns: accountid, enddate, isdeleted, systemmodstamp, startdate, status, createddate, contractnumber, contractterm, activateddate, sfid, id, _hc_lastop, and _hc_err. The main pane shows an SQL query window with the following code:

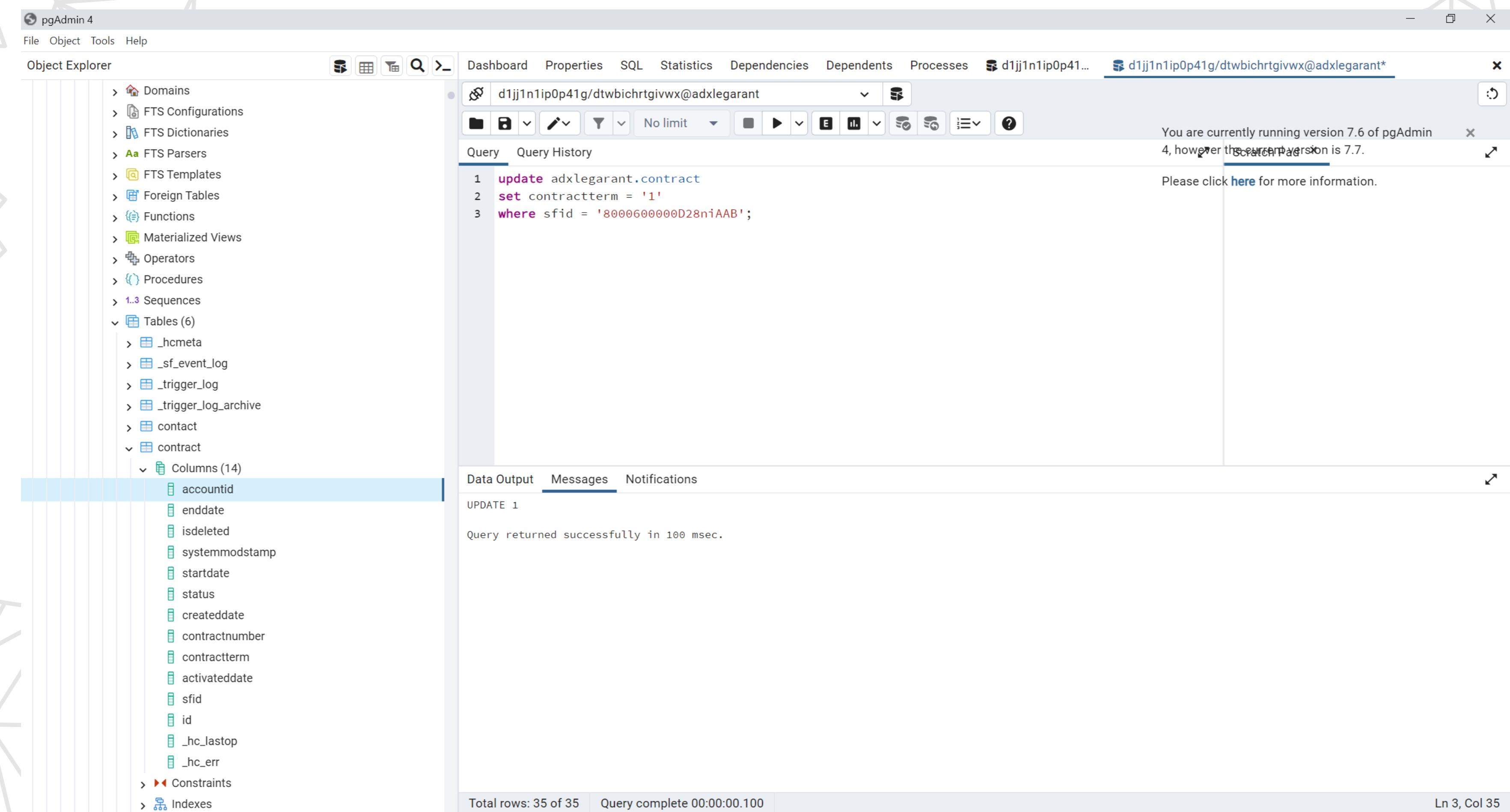
```
1 Insert into adxlegarant.contract (accountid, startdate, contractterm)
2 VALUES ('0010600002Cw3Z3AAJ', '12-12-2023', '6')
```

The message tab at the bottom indicates "Query returned successfully in 88 msec." and "Total rows: 35 of 35". A note in the top right corner says "You are currently running version 7.6 of pgAdmin 4, however the current page version is 7.7. Please click [here](#) for more information."



Environnement PGAdmin

Requêtes SQL Modification d'un contact



The screenshot shows the pgAdmin 4 interface. The left sidebar is titled "Object Explorer" and lists various database objects like Domains, FTS Configurations, FTS Dictionaries, etc., under "Tables (6)". A specific table named "contract" is selected, and its columns (accountid, enddate, isdeleted, systemmodstamp, startdate, status, createddate, contractnumber, contractterm, activateddate, sfid, id, _hc_lastop, _hc_err) are listed below it. The main pane contains a SQL query window with the following code:

```
1 update adxlegarant.contract
2 set contractterm = '1'
3 where sfid = '8000600000D28niAAB';
```

The "Messages" tab at the bottom shows the result of the query: "UPDATE 1" and "Query returned successfully in 100 msec.". A message in the top right corner indicates that the user is running version 7.6 of pgAdmin 4, while the current version is 7.7.





Déploiement de L'application

Deployer l'application

Repository
README.md

https://github.com/Magda-dev/Legrant_P12

Legrant Application Connectée Salesforce



How to deploy the app ?

Install the app package in your Salesforce org

Go to Salesforce and create you own regular org or sandbox. Then simply use the deploy to Salesforce org button to install the package on your own org and follow the 3 steps.

[Deploy to Salesforce](#)

Here are some documentation resources to get you started with Salesforce :

Configure Your Salesforce DX Project

The `sfdx-project.json` file contains useful configuration information for your project. See [Salesforce DX Project Configuration](#) in the [Salesforce DX Developer Guide](#) for details about this file.

Read more About Salesforce

- [Salesforce Extensions Documentation](#)
- [Salesforce CLI Setup Guide](#)
- [Salesforce DX Developer Guide](#)
- [Salesforce CLI Command Reference](#)

Install Postman

You can use the Postman desktop app or the Postman web UI to connect to Salesforce with the Salesforce APIs collection:

[Run in Postman](#)

[Video instructions](#)

[Download Postman](#)

[Install the Postman App](#)

[How to get started with Postman](#)

For instructions on how to install and set up Postman please visit :

[Video instructions](#)

- [Install the Postman App](#)
- [Fork the Collection](#)
- [Configure the Collection](#)
- [Authenticate with Salesforce](#)
- [Execute a Request](#)

See [additional documentation](#) for more information on how to keep the collection up to date and work with multiple Salesforce orgs.

Connect and deploy The app with Heroku in a few clicks

Sign in to a paid account on Heroku to get started. In this app Contact and Contract object are used and mapped in Heroku. Only Heroku Postgre (paid) and Heroku Connect add-ons are needed (Free version).

For simple deployment on Heroku platform use this handy Heroku-button

[Deploy to Heroku](#)

Command lines for Heroku

```
$ heroku create  
$ git push heroku main  
$ heroku open
```

Now you are all set to try out the app !





Magdalena Larmet

Salesforce Developer