Payment Gateway Implementation for ERP

Business Scenarios

- Admin (who ever has access to Fee Management Module) should be able to link bank accounts for each school.
  - There can be more than one account per school
  - Admin should have the facility to un-link/deactivate the linked account
  - There has to be at least one account that is active
- Admin should be able to generate a payment link for each fee per student and send it via SMS
- Payment links can be generated for any active account of the school (Multiple account cannot linke to one module)
- Payment links are generated specific to the student. End user (parent/gaurdian) who has the link can pay with a valid payment link
- Once the payment link is used to complete the payment, the link cannot be reused. The same link can be used if Payment is failed
- Payment gateway setup
  - There will be a base account for NTT (Currently TopScorer). School accounts will be linked to the NTT Razorpay Account. All Payments will be collected at NTT MID and then transferred via NTT to Schools MIDs/accounts

NOTE: Payment links should be linked to the Accounts directly. No need to write any code for this functionality. There is an API provided by Razorpay.

Out of scope

- A School account that is setup is no longer active at the Bank level. The payments would start failing. The system would not be able to check if the account is still active.

Points Discussed with Razorpay

1. There will be a base account for NTT (Currently TopScorrer) and the school accounts will be linked to the NTT Razorpay Account. All Payments made via NTT to Schools
2. As per the requirments, we will have max 5 active accounts per School
3. The settlement period is 1 day. Meaning, once parent pays the amount via Razorpay, the amount will be credited to NTT in 1 days
4. Razorpay supports to link the transfer account (School account) while initiating the payment
5. Unique Payment link can be generated per Student via RazorPay API
6. Deactivation of the School Account is a manual process. Email to be triggered to Razorpay to delete the account. SLA is not known yet (**Need Support Email of Razorpay – support@razorpay.com?**)
7. The default Expiry date for a payment link is 6 months.
8. Setting the Expiry date for a payment link is possible. The expiry date should be less than 6 months
9. Payment link will become invaild once the payment is done
10. There is a seperate notification mechanism in Razorpay which can be used to remind the parents for payment. Notification can be via SMS and/or Email.
11. No Seperate Charges to be paid to Razorpay for utilizing the notification mechanism
12. Maximum of 3 reminders can be set in Razorpay
13. Is there any dashboard to see the reminders sent to parents on Razorpay? **Need to confirm by Razorpay**
14. Currently Razorpay base account created for TopScorer. Needs to move the account to TopTech.
15. PO team to walk through the screen designs for Payment Gateway

Screen Flow

Step 1: Payment Gateway Landing Page

Step 2: Map Account Name with the below details

1. Business Name
2. Business Type (Below are possible values)
    a. Partnership
    b. Proprietorship
    c. Trust
    d. Individual
3. IFSC Code
4. Bank Name (populate Bank Name based on IFSC Code)
5. Branch Name (populate Bank Name based on IFSC Code)
6. Account Number
7. Beneficiary Name
8. Account Type

Step 3: Accept Terms & Conditions

Step 4: Manage list of all active accounts created in other modules. If not available, create a new account

Usecases / Functionalities

1. Razorpay Integration
2. Support of multiple linked account for each school
3. Personalization of payment link, including the details of the payment and the student
4. Create and send payment links (integration to notification service)
5. Configure Webhooks – successful or failed payment
6. Deactivate School Account
7. Support of Payment gateway for different ERP modules (build it in modular fashion to be able to provide support to all future modules of ERP)

Auditing Requirements

1. Capture the user details and timestamp
    a. On Account creation and updation
    b. while adding the Account to the Module
2. Capture Module name and user information who is requesting for the payment link generation
3. Capture every attempt made by the user on the payment link (fail and success)
4. Use “System” as a user whenever a payment link request is triggered by the MQ/Cron Job

DB Structure

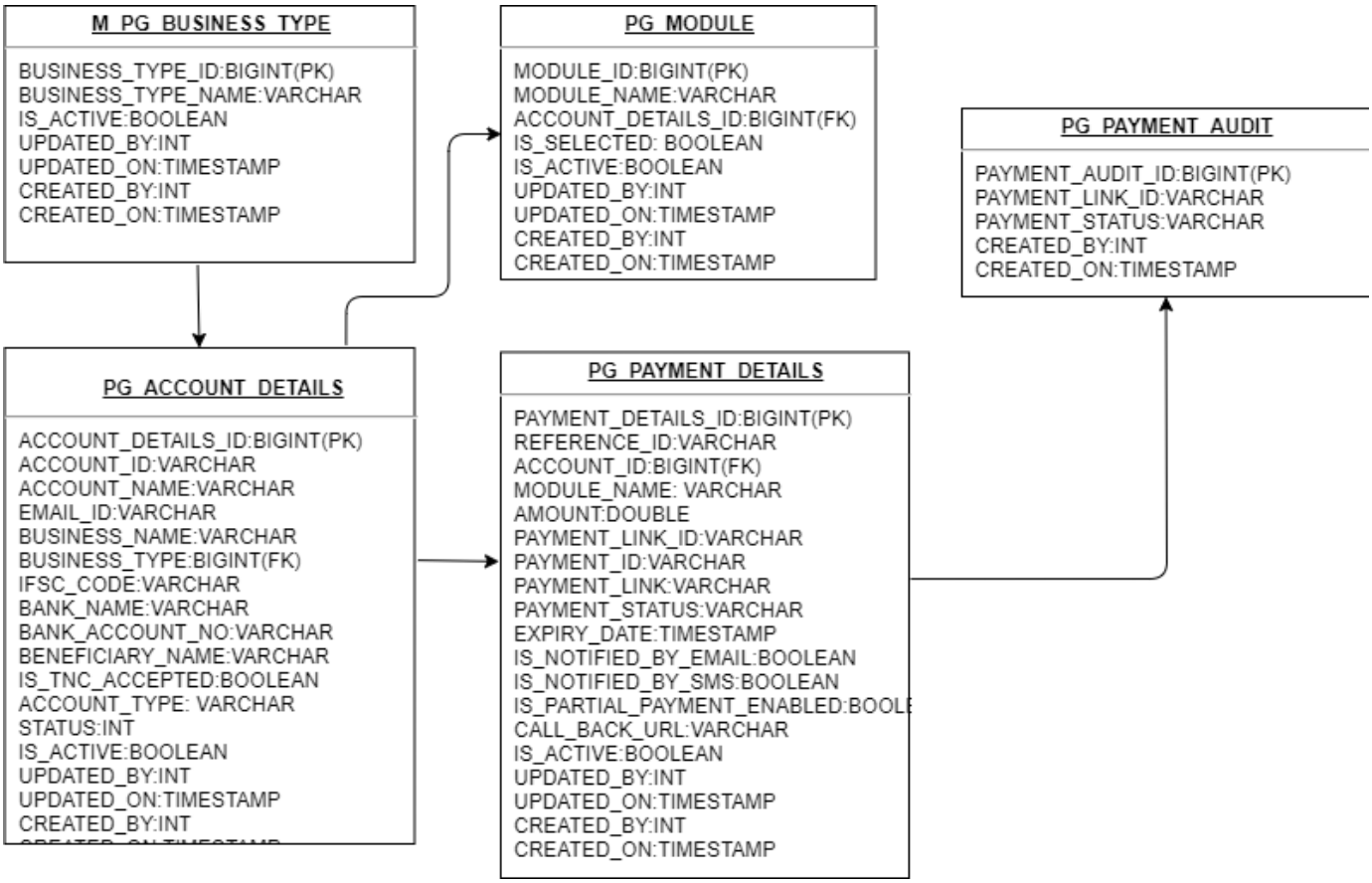


Table 1: **eSense.M_PG_BUSINESS_TYPE**

Master lookup table for all the business types

COLUMN	DATA TYPE	SIZE	DEFAULT	DESCRIPTION
BUSINESS_TYPE_ID	PK			PRIMARY KEY OF THE BUSINESS TYPE
BUSINESS_TYPE_NAME	VARCHAR	100		NAME OF THE BUSINESS TYPE
IS_ACTIVE	BOOLEAN		TRUE	FLAG FOR SOFT DELETE
UPDATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO UPDATED THE RECORD RECENTLY
UPDATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER UPDATED THE RECORD RECENTLY
CREATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO CREATED THE RECORD
CREATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER CREATED THE RECORD

Table 2: **ERP.PG_ACCOUNT_DETAILS**

Entry of all the accounts that can be configured for the School. This would be in the school specific schema

COLUMN	DATA TYPE	SIZE	DEFAULT	DESCRIPTION
ACCOUNT_DETAILS_ID	BIGINT		SERIAL	PRIMARY KEY
ACCOUNT_ID	VARCHAR	30	UNIQUE	RAZORPAY ACCOUNT ID
ACCOUNT_NAME	VARCHAR	50		Name of the account holder
EMAIL_ID	VARCHAR	50		Email address of the account holder
BUSINESS_NAME	VARCHAR	50		Name of the School
BUSINESS_TYPE	FK			Type of the School

IFSC_CODE	VARCHAR	11		The ifsc code of the bank account
BANK_NAME	VARCHAR	50		Name of the bank account holder
BANK_ACCOUNT_NO	VARCHAR	30		The bank account number
BENEFICIARY_NAME	VARCHAR	50		Name of the bank account holder
IS_TNC_ACCEPTED	BOOLEAN		FALSE	Indicates if the terms and conditions have been accepted or not
ACCOUNT_TYPE	VARCHAR	20		Type of the bank account. Savings or Current
STATUS	INT		FK	Status of the account
IS_ACTIVE	BOOLEAN		TRUE	FLAG FOR SOFT DELETE
UPDATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO UPDATED THE RECORD RECENTLY
UPDATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER UPDATED THE RECORD RECENTLY
CREATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO CREATED THE RECORD
CREATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER CREATED THE RECORD

Table 3: **ERP.PG_MODULE**

Module to bank account mapping would be managed in this table

COLUMN	DATA TYPE	SIZE	DEFAULT	DESCRIPTION
MODULE_ID	BIGINT		PK	SERIAL
MODULE_NAME	VARCHAR	20		Name of the module (e.g: FMS, ADM etc)
ACCOUNT_ID	BIGINT		FK	Account reference of PG_ACCOUNT_DETAILS
IS_ACTIVE	BOOLEAN		TRUE	FLAG FOR SOFT DELETE
UPDATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO UPDATED THE RECORD RECENTLY
UPDATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER UPDATED THE RECORD RECENTLY
CREATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO CREATED THE RECORD
CREATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER CREATED THE RECORD

Table 4: **ERP.PG_PAYMENT_DETAILS**

The payment links generated for all purposes per student will be stored.

COLUMN	DATA TYPE	SIZE	DEFAULT	DESCRIPTION
PAYMENT_DETAILS_ID	BIGINT		SERIAL	Primary Key of the table
REFERENCE_ID	VARCHAR			Unique ID for the reference of the payment, generated with a combination of unique text and tenant ID
ACCOUNT_ID	BIGINT		FK	Account reference of PG_ACCOUNT_DETAILS
AMOUNT	DOUBLE			Amount paid
PAYMENT_LINK_ID	VARCHAR	30	UNIQUE	Unique Payment Link ID generated by Razorpay
PAYMENT_ID	VARCHAR	30	UNIQUE	Payment ID of the successful payment
PAYMENT_LINK	VARCHAR	100		Payment link generated by Razorpay
PAYMENT_STATUS	VARCHAR	20		Payment Status
EXPIRY_DATE	TIMESTAMP		NOT NULL	Expiry date of the payment link
IS_NOTIFIED_BY_EMAIL	BOOLEAN		FALSE	Is_Notified_By Reference of Is_Notified_By table
IS_NOTIFIED_BY_SMS	BOOLEAN		FALSE	Is_Notified_By Reference of Is_Notified_By table

CALL_BACK_URL	VARCHAR	500		Unique Call Back URL generated by Razorcal
MODULE_NAME	VARCHAR	100		The name of the module from which the request has come to create the payment link
IS_ACTIVE	BOOLEAN		TRUE	FLAG FOR SOFT DELETE
UPDATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO UPDATED THE RECORD RECENTLY
UPDATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER UPDATED THE RECORD RECENTLY
CREATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO CREATED THE RECORD
CREATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER CREATED THE RECORD

Table 5: **ERP.M_PG_PAYMENT_AUDIT**

Audit table to track the payment on every event on the Payment (e.g: Paid, expired, cancelled etc)

COLUMN	DATA TYPE	SIZE	DEFAULT	DESCRIPTION
PAYMENT_AUDIT_ID	PK			PRIMARY KEY OF THE TABLE
PAYMENT_LINK_ID	VARCHAR	30	UNIQUE	Unique Payment Link ID generated by Razorpay
PAYMENT_STATUS	VARCHAR	20		Payment status of a payment link generated by Razorpay
CREATED_BY	INT		NOT NULL	AUDIT COLUMN - USER WHO CREATED THE RECORD
CREATED_ON	TIMESTAMP		CURRENT TIMESTAMP	AUDIT COLUMN - DATE & TIME WHEN USER CREATED THE RECORD

Razorpay Integration

Razorpay is a payment gateway service provider authorises credit card / UPI / Netbanking / Wallet payments.

Setup Razorpay with ERP

Step 1: To enable and integrate RazorPay payment service, we need to configure API Key, API Secret parameters under Dashboard Settings API Keys

Step 2: Add below dependency in pom.xml

```
<!-- https://mvnrepository.com/artifact/com.razorpay/razorpay-java  -->
<dependency>
    <groupId>com.razorpay</groupId>
    <artifactId>razorpay-java</artifactId>
    <version>1.4.3</version>
</dependency>
```

Step 3: Configure Razorpay API Keys (Key & Secret) in ERP Config Server

```
razorpay.key={key}
razorpay.secret={secret}
```

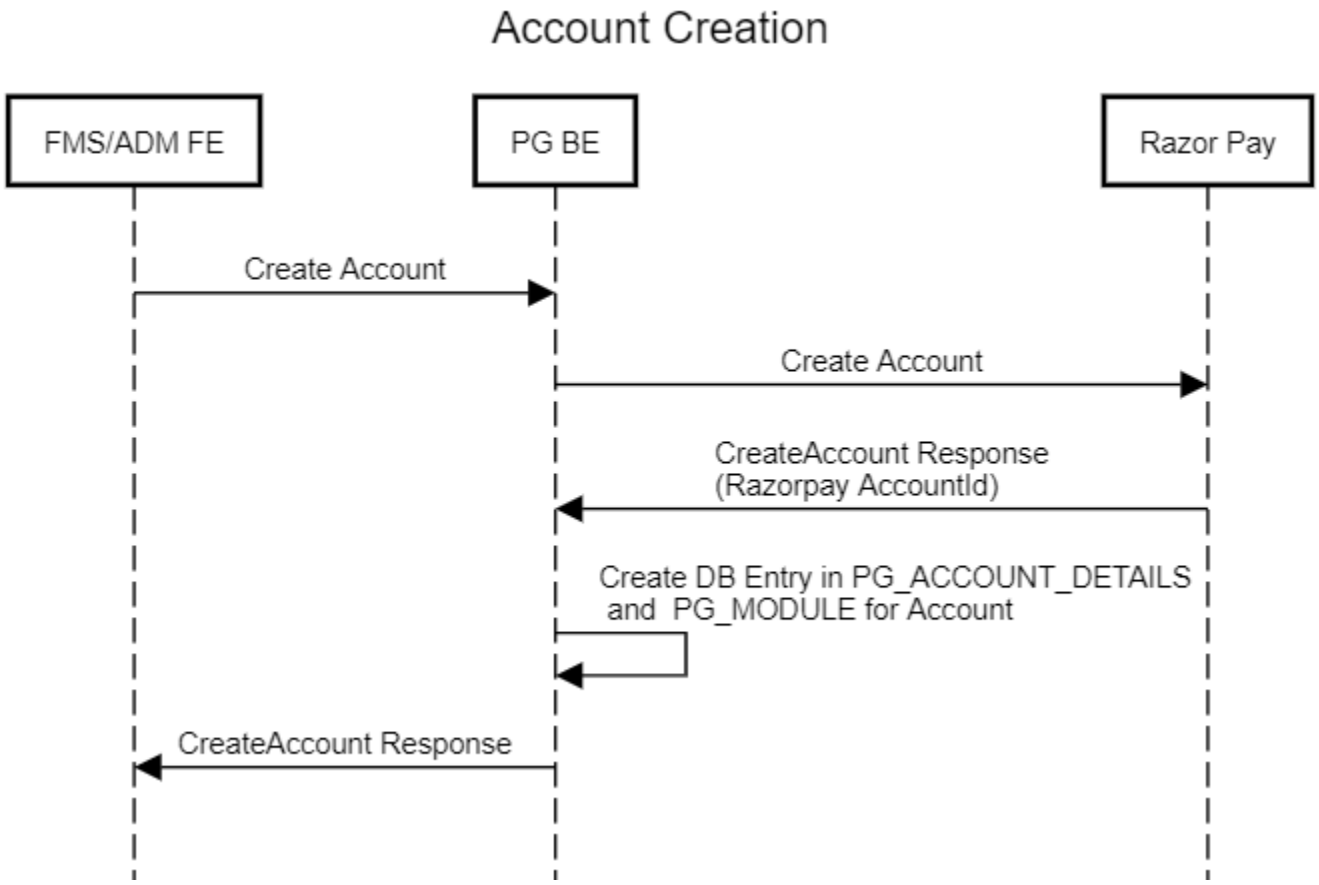
Step 4: Razorpay Client Configuration

```
import org.springframework.boot.context.properties.
ConfigurationProperties;
import org.springframework.stereotype.Component;

import lombok.Data;

@Data
@Component
@ConfigurationProperties(prefix = "razorpay")
public class RazorPayClientConfig {
    private String key;
    private String secret;
}
```

Flows

Flow 1: Account Creation Flow

Below is the flow triggered once the user clicks on Finish button on Account Creation Page



IFSC CODE Validation

IFSC CODE should be validate with the below API

API URL: https://ifsc.razorpay.com/{ifsccode}

HTTP Method: GET

Response Type: Json

Sample Response

```
URL: https://ifsc.razorpay.com/YESB0DNB002
{
        "BRANCH": "Delhi Nagrik Sehkari Bank IMPS",
        "CENTRE": "DELHI",
        "DISTRICT": "DELHI",
        "STATE": "MAHARASHTRA",
        "ADDRESS": "720, NEAR GHANTAGHAR, SUBZI MANDI, DELHI - 110007",
        "CONTACT": "+919560344685",
        "IMPS": true,
        "CITY": "MUMBAI",
        "UPI": true,
        "MICR": "110196002",
        "RTGS": true,
        "NEFT": true,
        "SWIFT": "",
        "ISO3166": "IN-MH",
        "BANK": "Delhi Nagrik Sehkari Bank",
        "BANKCODE": "DENS",
        "IFSC": "YESB0DNB002""
}
```

API Details

ERP Request Payload:

Request HTTP Method: POST

API End Point: `payment-gateway/accounts/save`

```
{
    "AccountName": "Blue Bell Account 1",
    "email": "bluebell@gmail.com",
    "businessName": "Blue Bell",
    "businessType": "3",
    "ifscCode": "ABHY0065306",
    "bankName": "ABYUDAYA COOP BANK",
    "accountNo": "12892345789123456",
    "accountType": "current",
    "beneficiaryName": "Gabriel A. Jackson",
    "tnc": true
}
```

Response Payload (Success):


```
{
"data": [
    "accountId": "acc_gHQwerty123ggd",
    "AccountName": "Blue Bell Account 1",
    "email": "bluebell@gmail.com",
    "businessName": "Blue Bell",
    "businessType": "3",
    "ifscCode": "ABHY0065306",
    "bankName": "ABYUDAYA COOP BANK",
    "accountNo": "12892345789123456",
    "beneficiaryName": "Gabriel A. Jackson",
    "accountType": "current",
    "tnc": true,
    "status": "activated",
    "updatedBy": "1",
    "updatedOn": "2022-11-15T18:30:00.000Z",
    "createdBy": "1",
    "createdOn": "2022-11-15T18:30:00.000Z"
  ],
   "result": {
        "responseCode": 201,
        "responseDescription": "Success"
    }
}
```


Response Payload (Failure):


```
{
    "timestamp": "2022-09-14T13:34:26.310+00:00",
    "status": 500,
    "error": "Internal Server Error",
    "path": "/fm/api/v1/payment-gateway/createAccount"
}
```


Razorpay API:

`https://api.razorpay.com/v1/beta/accounts`

Razorpay API Documentation: https://razorpay.com/docs/api/payments/route/account-apis-beta/

Sample Razorpay API request for Account Creation

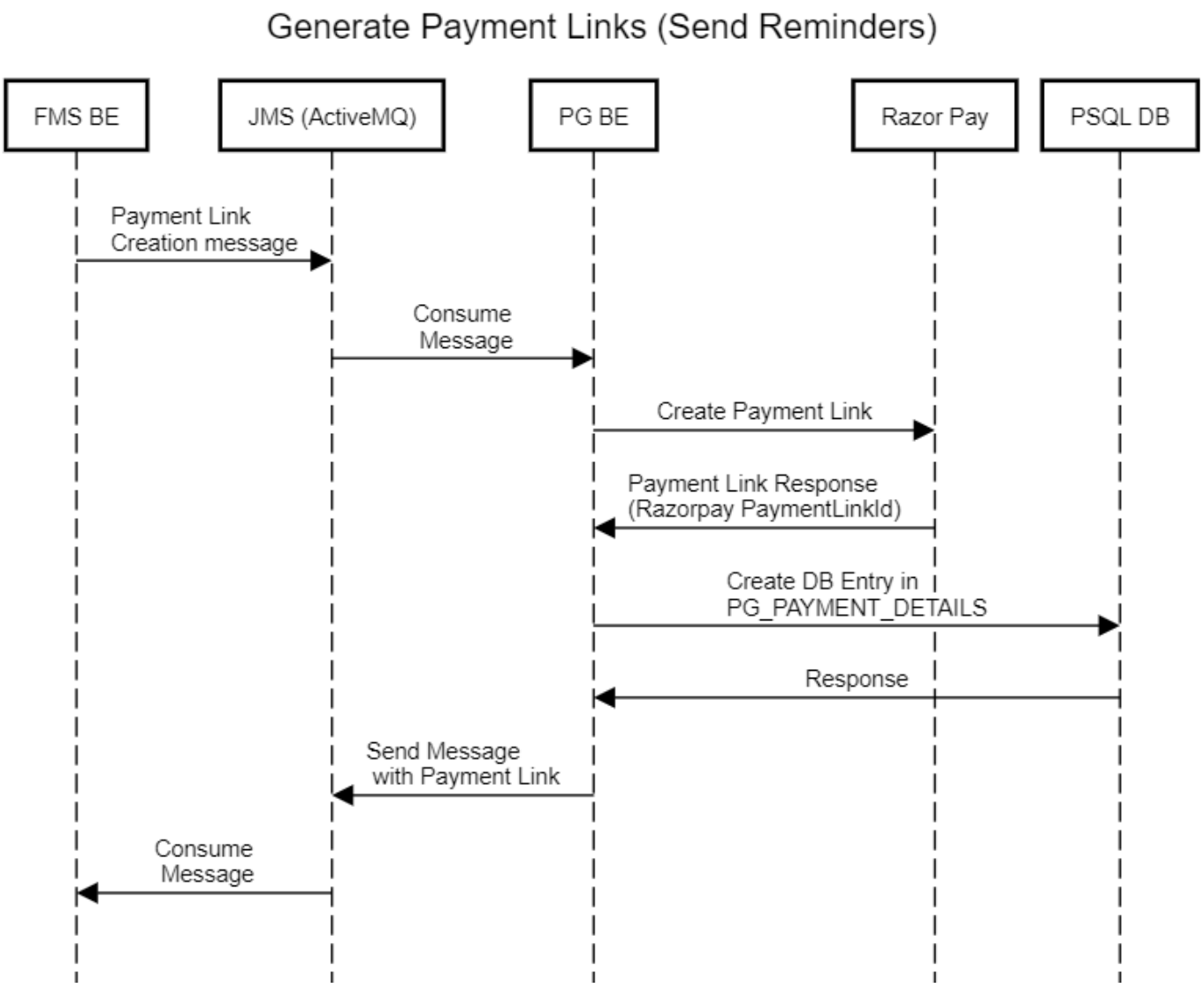
```
curl -XPOST 'https://api.razorpay.com/v1/beta/accounts' \
     -u [YOUR_KEY_ID]:[YOUR_SECRET] \
     -H "Content-type: application/json" \
     -d '{
           "name":"Gaurav Kumar",
           "email":"gaurav.kumar@example.com",
           "tnc_accepted":true,
           "account_details":{
              "business_name":"Acme Corporation",
              "business_type":"Trust"
           },
           "bank_account":{
              "ifsc_code":"HDFC0CAGSBK",
              "beneficiary_name":"Gaurav Kumar",
              "account_type":"current",
              "account_number":1234567890123456
           }
        }'
```

Sample Java Code for Account Creation

```
@Autowired
private RazorPayClientConfig razorPayClientConfig;

RazorpayClient razorpay = new RazorpayClient(razorPayClientConfig.
getKey(), razorPayClientConfig.getSecret());
JSONObject accountCreationRequest = new JSONObject();
accountCreationRequest.put("name","Gaurav Kumar");
accountCreationRequest.put("email","gaurav.kumar@example.com");
accountCreationRequest.put("tnc_accepted",true);

JSONObject accountDetails = new JSONObject();
accountDetails.put("business_name","Acme Corporation");
accountDetails.put("business_type","Trust");
accountCreationRequest.put("account_details",accountDetails)

JSONObject bankAccount = new JSONObject();
bankAccount.put("ifsc_code","HDFC0CAGSBK");
bankAccount.put("beneficiary_name","Gaurav Kumar");
bankAccount.put("account_type","current");
bankAccount.put("account_number","1234567890123456");

HttpHeaders headers = new HttpHeaders();
headers.set(key, apiKeyValue);
headers.set(secret, secretValue);
headers.setContentType(MediaType.APPLICATION_JSON);
headers.setAccept(Collections.singletonList(MediaType.
APPLICATION_JSON));
HttpEntity<String> request = new HttpEntity<String>
(accountCreationRequest.toString(), headers);
String accountResultAsJsonStr =
      restTemplate.postForObject("https://api.razorpay.com/v1/beta
/accounts", request, String.class);
JsonNode root = objectMapper.readTree(accountResultAsJsonStr);
```

Flow 2: Send Reminders (Payment Link Generation) Flow

Below is the flow triggered once the user clicks on Send Reminders on Fee Collection and Defaulters



The below fields to be sent to Active MQ for payment link creation

```
studentId
gradeId
amount
description
expiryDate
moduleName
```

Once Payment link is created by Payment gateway service. Below fields will be sent to the Active MQ

```
studentId
gradeId
amount
description
expiryDate
moduleName
paymentLinkId
paymentLink
paymentStatus
```

Razorpay API for Payment Links:

`https://api.razorpay.com/v1/payment_links`

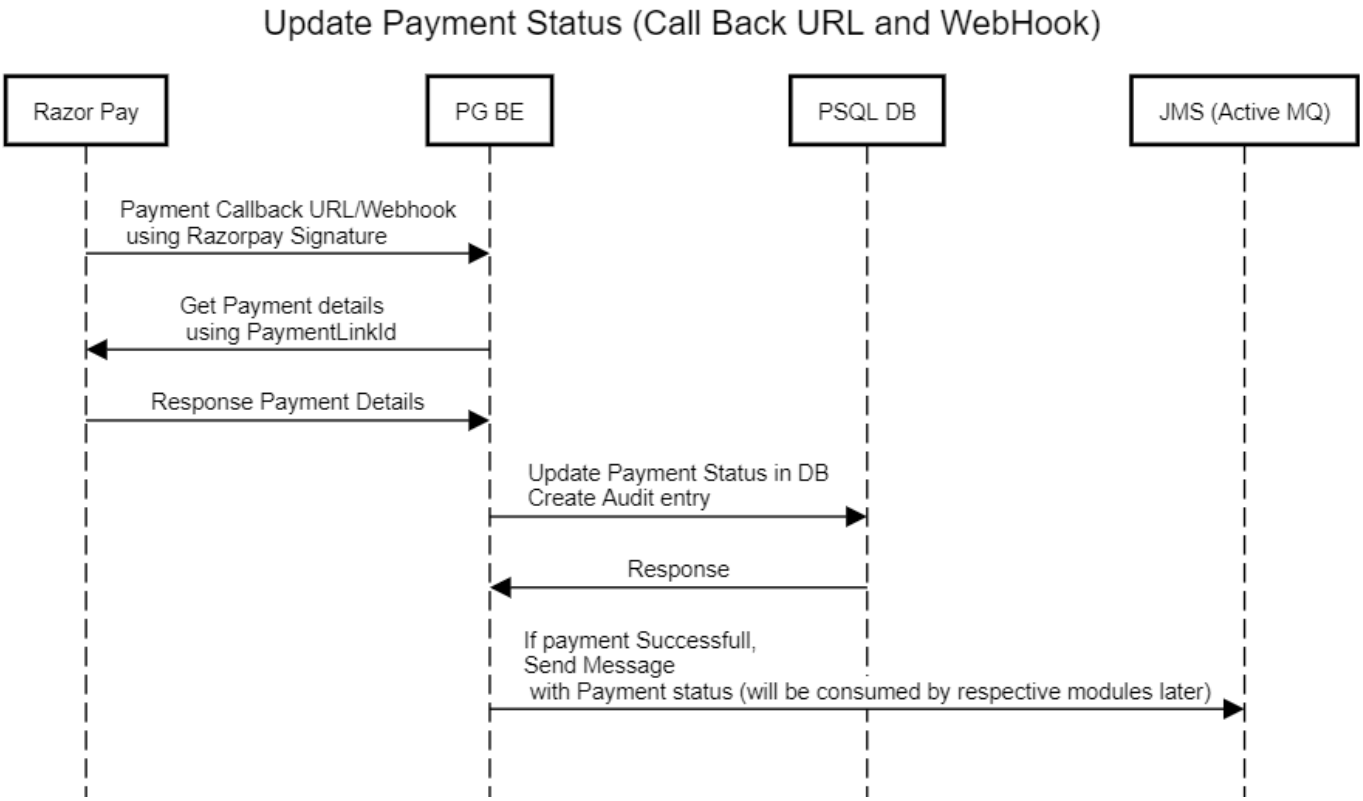
Razorpay API Documentation: https://razorpay.com/docs/api/payments/payment-links/

Sample Razorpay API request for Account Creation

```
curl -u [YOUR_KEY_ID]:[YOUR_KEY_SECRET] \
-X POST https://api.razorpay.com/v1/payment_links/ \
-H 'Content-type: application/json' \
-d '{
  "amount": 1000,
  "currency": "INR",
  "accept_partial": true,
  "first_min_partial_amount": 100,
  "expire_by": 1691097057,
  "reference_id": "TS1989",
  "description": "Payment for policy no #23456",
  "customer": {
    "name": "Gaurav Kumar",
    "contact": "+919999999999",
    "email": "gaurav.kumar@example.com"
  },
  "notify": {
    "sms": true,
    "email": true
  },
  "reminder_enable": true,
  "notes": {
    "policy_name": "Jeevan Bima"
  },
  "callback_url": "https://example-callback-url.com/",
  "callback_method": "get"
}'
```

Flow 3: Payment Status Update (Callback / Webhook) Flow

Below is the flow triggered once the user completes the Payment



Callback URL: {baseURL}/payment-gateway/paymentlink/callback/webhook?razorpay_payment_id={paymentId}&razorpay_payment_link_id={paymentLinkId}&razorpay_payment_link_reference_id={referenceId}&razorpay_payment_link_status={paidStatus}&razorpay_signature={signatureKey}&tenantId=&tenantId

NOTE: Request Parameters are appended by the Razorpay to the callback URL

Verify Signature

You can verify the `razorpay_signature` parameter to validate that it is authentic and sent from Razorpay servers.

- The `razorpay_payment_link_id` attribute should be stored in your system against an order, right after it is returned in the create Payment Link response. This is displayed as just `id` (for example, "id": "plink_FKeEiabyAAiSVQ") in the response.
- The `razorpay_signature` should be validated by your server. In order to verify the signature, you need to create a signature using
  - `razorpay_payment_link_id`
  - `razorpay_payment_link_reference_id`
  - `razorpay_payment_link_status`
  - `razorpay_payment_id`as payload and your `key_secret` (your API secret) as secret.

```
RazorpayClient razorpay = new RazorpayClient("[YOUR_KEY_ID]",
"[YOUR_KEY_SECRET]");

String secret = "EnLs21M47BllR3X8PSFtjtbd";

JSONObject options = new JSONObject();
options.put("payment_link_reference_id", "TSsd1989");
options.put("razorpay_payment_id", "pay_IH3d0ara9bSsjQ");
options.put("payment_link_status", "paid");
options.put("payment_link_id", "plink_IH3cNucfVEgV68");
options.put("razorpay_signature",
"07ae18789e35093e51d0a491eb9922646f3f82773547e5b0f67ee3f2d3bf7d5b");

boolean status =  Utils.verifyPaymentLink(options, secret);
```

After validating the signature, you should fetch the order in your system corresponding to the `razorpay_payment_link_id` and mark this order as successful.

Webhooks

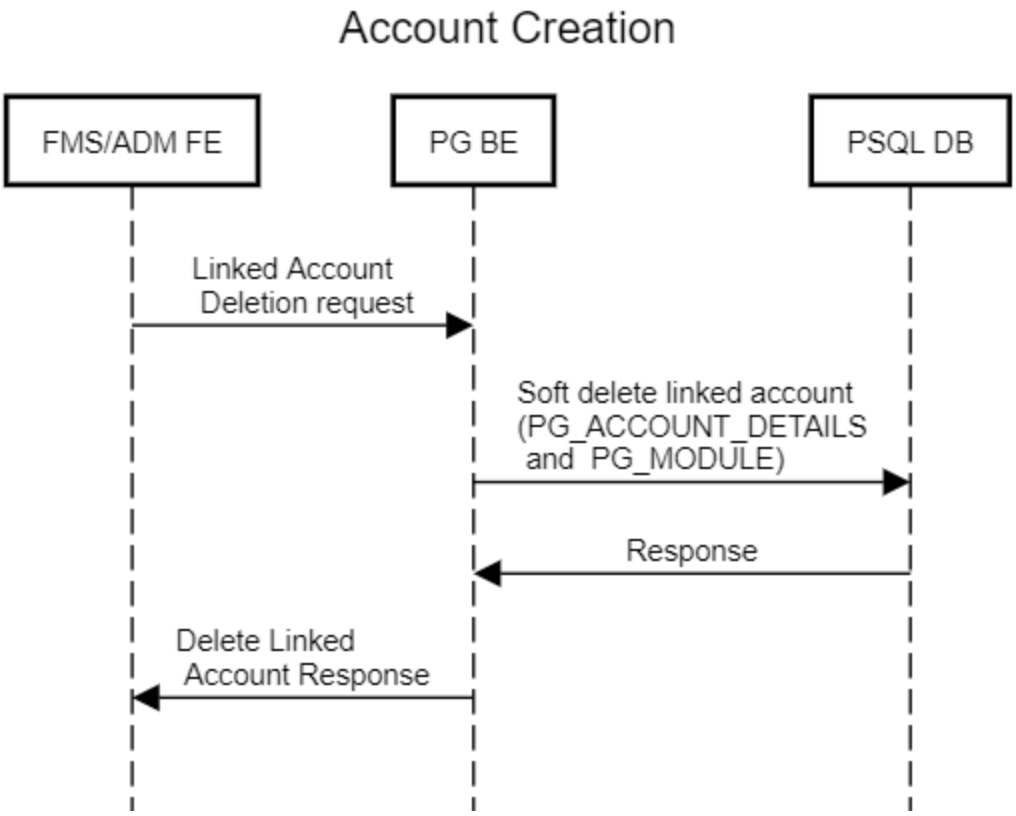
Webhooks are one way that Razorpay can communicate with the application in real-time when a specific event happens like Payments, orders, settlements etc. In our case, when the payment is success / fail this events is triggered and Razorpay send an HTTP POST payload in JSON to the webhook's configured URL.

To have a fail safe mechanism, we configure the below URL on the Razorpay as a webhook on the events **payment_link.paid, payment_link.partially_paid, payment_link.expired, payment_link.cancelled.**

{baseURL}/payment-gateway/paymentlink/webhook

Flow 4: Delete Linked Account Flow

Below is the flow triggered once the user Deletes the linked account



API Details

Request Payload:

Request HTTP Method: GET

API End Point: `payment-gateway/accounts/deleteAccount/{accountDetailsId}`

Response Payload (Success):

```
{
"data": [
    "message": "Account Deactiviated Successfully"
  ],
   "result": {
        "responseCode": 201,
        "responseDescription": "Success"
    }
}
```

Response Payload (Failure):

```
{
    "timestamp": "2022-09-14T13:34:26.310+00:00",
    "status": 500,
    "error": "Internal Server Error",
    "path": "/fm/api/v1/payment-gateway/deactivateAccount"
}
```

References

Razorpay Documentation / Developer Guide : https://razorpay.com/docs/api/

Figma Screens: https://www.figma.com/file/SQ30MNT1KMGLZWfMvmDb8I/Fee-Management?node-id=11496%3A125267

Team Structure

Developer	No. of Developers	Dev Estimation	Start Date	End Date
FE Developer	2	181	20-Sep-2022	19-Oct-2022
BE Developer	1	126	19-Sep-2022	19-Oct-2022

Plan - TO BE FINALIZED

Release 1 – 05th Oct

Demo 1 – 10th Oct

Release 2 – 12th Oct

Demo 2 – 17th Oct

Release 3 – 19th Oct

Demo 3 – 21th Oct or 28th Oct

JIRA ID	User story	Dev End Date	Release
EE-833	Payment gateway onboarding landing page - Provision to setup new payment gateway	27-Sep	1
EE-871	Payment gateway onboarding landing page - Add Account	28-Sep	1
EE-872	Payment gateway onboarding landing page - Add Bank Account Details	06-Oct	2
EE-888	Payment gateway onboarding landing page - Terms and Conditions	10-Oct	2
EE-939	Payment gateway onboarding landing page - View Details of added account	04-Oct	2
EE-944	Payment gateway onboarding landing page - Select any existing account	06-Oct	2
EE-945	Payment gateway onboarding landing page - 'Skip and Finish' and 'Finish' button	13-Oct	3
EE-948	Payment gateway post setup - landing page	11-Oct	2
EE-949	Payment gateway post setup - Delete Account	14-Oct	3
EE-952	Payment gateway post setup - Setup payment gateway	14-Oct	3
EE-956	Payment gateway post setup - Select any existing account	18-Oct	3
EE-957	Payment gateway post setup - Add New Account	17-Oct	3
EE-1021	Generation of Payment Links	19-Oct	3

EE-1022	Integration of Payment Links with Send Reminders	20-Oct	3
EE-1025	Integration of Payment Gateway Component with Fee Management UI	19-Oct	4