Pg restore command for **pg_dump file :**

psql -U postgres -d tnelb_staging -p 5555 -f "C:\xampp\htdocs\TNelb-Staging\db\23102025\tnelb_backup_plain.sql"

List database **:**

Psql -U postgres -f

```
CREATE TABLE IF NOT EXISTS public.tnelb_forms
(
    f_id integer NOT NULL DEFAULT nextval('tnelb_forms_f_id_seq'::regclass),
    form_name character varying(50) COLLATE pg_catalog."default" NOT NULL,
    license_name character varying(50) COLLATE pg_catalog."default",
    cert_licence_code character varying(50) COLLATE pg_catalog."default",
    form_code character varying(50) COLLATE pg_catalog."default",
    fresh_fee_amount bigint,
    fresh_fee_starts date,
    fresh_fee_ends date,
    renewal_amount bigint,
    renewalamount_starts date,
    renewalamount_ends date,
    latefee_amount integer,
    latefee_starts date,
    latefee_ends date,
    duration_freshfee integer,
    duration_freshfee_starts date,
    duration_freshfee_ends date,
    duration_renewalfee integer,
    duration_renewalfee_starts date,
    duration_renewalfee_ends date,
    duration_latefee integer,
    duration_latefee_starts date,
    duration_latefee_ends date,
    instructions_upload text COLLATE pg_catalog."default",
    category integer,
    status integer,
    created_by character varying(255) COLLATE pg_catalog."default",
    updated_by character varying(255) COLLATE pg_catalog."default",
    created_at timestamp(0) with time zone DEFAULT now(),
    updated_at timestamp(0) without time zone,
    CONSTRAINT tnelb_forms_pkey PRIMARY KEY (f_id)
)
```

TABLESPACE pg_default;

```
select * from tnelb_fees where cert_licence_id = '1' AND fees_type = 'R' AND
start_date <= now() order by
start_date desc LIMIT 1

select * from mst_fees_validity where form_type = 'L' AND licence_id = '1' AND
validity_start_date <= now()
order by validity_start_date desc LIMIT 1


SELECT EXTRACT(MONTH FROM AGE('2025-11-11', '2025-12-11')) AS
month_difference;



CREATE OR REPLACE FUNCTION test_late_fee_full_final()
RETURNS TABLE (
   expiry_date date,
   today_date date,
   validity_months int,
   month_diff int,
   late_fee numeric,
   base_fee numeric,
   total_fee numeric,
   fee_type varchar(1),
   status text
) AS $$
DECLARE
   v_expiry_date date := '2025-05-01';     -- expiry date
   v_today_date date := '2025-06-01';      -- today date (test)
   v_validity_months int := 3;             -- validity before expiry
   v_late_fee_per_month numeric := 300;    -- per month
   v_base_fee numeric := 1500;             -- renewal fee
   v_fresh_fee numeric := 2500;            -- fresh fee
   v_month_diff int := 0;
   v_late_fee numeric := 0;
   v_total_fee numeric := 0;
BEGIN
   --1️⃣ Calculate month difference
   v_month_diff := (DATE_PART('year', age(v_expiry_date, v_today_date)) * 12)
                + DATE_PART('month', age(v_expiry_date, v_today_date));

   --2️⃣ Normal renewal
   IF v_today_date < (v_expiry_date - (v_validity_months || ' month')::interval) THEN
      v_late_fee := 0;
      v_total_fee := v_base_fee;
```

```
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_fee, v_base_fee, v_total_fee, 'R'::varchar(1),
                    'NORMAL RENEWAL - NO LATE FEE';

    --3️⃣ Within 3 months before expiry
    ELSIF v_today_date BETWEEN (v_expiry_date - (v_validity_months || '
month')::interval) AND v_expiry_date THEN
        v_late_fee := ((v_validity_months - v_month_diff) + 1) * v_late_fee_per_month;
        IF v_late_fee > (v_validity_months * v_late_fee_per_month) THEN
            v_late_fee := v_validity_months * v_late_fee_per_month;
        END IF;
        v_total_fee := v_base_fee + v_late_fee;
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_fee, v_base_fee, v_total_fee, 'R'::varchar(1),
                    'LATE FEE APPLICABLE (BEFORE EXPIRY)';

    --4️⃣ After expiry but within 1 year
    ELSIF v_today_date > v_expiry_date AND v_today_date <= (v_expiry_date +
INTERVAL '1 year') THEN
        v_late_fee := v_validity_months * v_late_fee_per_month;
        v_total_fee := v_base_fee + v_late_fee;
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_fee, v_base_fee, v_total_fee, 'R'::varchar(1),
                    'LATE FEE (AFTER EXPIRY - WITHIN 1 YEAR)';

    --5️⃣ After 1 year → fresh application
    ELSE
        v_late_fee := 0;
        v_total_fee := v_fresh_fee;
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_fee, v_fresh_fee, v_total_fee, 'N'::varchar(1),
                    'FRESH APPLICATION (AFTER 1 YEAR)';
    END IF;
END;
$$ LANGUAGE plpgsql;



select * from test_late_fee_full_final();




CREATE OR REPLACE FUNCTION test_late_fee_simplified()
```

```
RETURNS TABLE (
    expiry_date date,
    today_date date,
    validity_months int,
    month_diff int,
    late_months int,
    late_fee numeric,
    base_fee numeric,
    total_fee numeric,
    fee_type varchar(1),
    status text
) AS $$
DECLARE
    v_expiry_date date := '2025-05-01';     -- static expiry date
    v_today_date date := '2025-02-01';      -- static current date
    v_validity_months int := 3;             -- validity period (3 months)
    v_late_fee_per_month numeric := 300;    -- late fee per month
    v_base_fee numeric := 1500;             -- renewal fee
    v_month_diff int := 0;
    v_late_months int := 0;
    v_late_fee numeric := 0;
    v_total_fee numeric := 0;
BEGIN
    -- 🧮1️⃣Calculate month difference
    v_month_diff := EXTRACT(MONTH FROM AGE(v_expiry_date, v_today_date));

    -- 🧮2️⃣Calculate late months = validity - difference
    v_late_months := v_validity_months - v_month_diff;

    -- 🧠 Case 1: Before the late-fee validity period → no late fee
    IF v_late_months <= 0 THEN
        v_late_fee := 0;
        v_total_fee := v_base_fee;
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_months, v_late_fee, v_base_fee, v_total_fee,
                    'R'::varchar(1), 'NORMAL RENEWAL - NO LATE FEE';

    -- 🧠 Case 2: Within late-fee validity (before expiry)
    ELSE
        v_late_fee := v_late_months * v_late_fee_per_month;
        v_total_fee := v_base_fee + v_late_fee;
        RETURN QUERY SELECT v_expiry_date, v_today_date, v_validity_months,
v_month_diff,
                    v_late_months, v_late_fee, v_base_fee, v_total_fee,
                    'R'::varchar(1), 'LATE FEE APPLICABLE (BEFORE EXPIRY)';
    END IF;
END;
```

```
$$ LANGUAGE plpgsql;



select * from test_late_fee_simplified();









SELECT
   mst_licences.licence_name,
   mst_licences.form_name,
   tnelb_fees.*,
   CASE
      WHEN tnelb_fees.start_date <= '2025-11-10' THEN 'Active'
      ELSE 'Inactive'
   END AS status
FROM
   tnelb_fees
LEFT JOIN
   mst_licences
   ON tnelb_fees.cert_licence_id = mst_licences.id
Where
         tnelb_fees.cert_licence_id = 1
ORDER BY
   CASE
      WHEN tnelb_fees.start_date <= '2025-11-10' THEN 1
      ELSE 2
   END,
   tnelb_fees.start_date DESC;
```

Lates function

```
-- FUNCTION: public.calc_late_fee_dynamic(character varying, integer, character
varying)

-- DROP FUNCTION IF EXISTS public.calc_late_fee_dynamic(character varying,
integer, character varying);

CREATE OR REPLACE FUNCTION public.calc_late_fee_dynamic(
        p_form_type character varying,
        p_licence_id integer,
        p_issued_licence_no character varying DEFAULT NULL::character varying)
   RETURNS TABLE(expiry_date date, today_date date, validity_months integer,
month_diff integer, late_months integer, late_fee numeric, base_fee numeric, total_fee
numeric, status text)
   LANGUAGE 'plpgsql'
   COST 100
   VOLATILE PARALLEL UNSAFE
   ROWS 1000

AS $BODY$
DECLARE
   v_today_date DATE := CURRENT_DATE;
   v_expiry_date DATE := NULL;
   v_validity_months INT := 0;
   v_late_fee_per_month NUMERIC := 0;
   v_base_fee NUMERIC := 0;
   v_fresh_fee NUMERIC := 0;
   v_month_diff INT := 0;
   v_late_months INT := 0;
   v_late_fee NUMERIC := 0;
   v_total_fee NUMERIC := 0;
BEGIN
   -- 🔹 1️⃣ Fetch expiry when Renewal (R)
   IF p_form_type = 'R' THEN
      IF p_issued_licence_no IS NULL THEN
         RAISE EXCEPTION 'issued_licence_no must be provided for Renewal
(form_type = R)';
      END IF;

      SELECT expires_at
      INTO v_expiry_date
      FROM tnelb_license
      WHERE license_number = p_issued_licence_no
      LIMIT 1;
```

```
        IF v_expiry_date IS NULL THEN
            RAISE EXCEPTION 'No record found for issued_licence_no: %',
p_issued_licence_no;
        END IF;
    END IF;

    -- 🔹 2️⃣ Get base fee from tnelb_fees
    SELECT f.fees
    INTO v_base_fee
    FROM tnelb_fees f
    WHERE f.cert_licence_id = p_licence_id
      AND f.fees_type = p_form_type
      AND f.start_date <= CURRENT_DATE
    ORDER BY f.start_date DESC
    LIMIT 1;

    IF NOT FOUND THEN
        RAISE NOTICE 'No fee configuration found for licence_id % and form_type %',
p_licence_id, p_form_type;
        RETURN;
    END IF;

    -- 🧮 3️⃣ For non-renewal (like F or N)
    IF p_form_type <> 'R' THEN
        RETURN QUERY
        SELECT NULL::DATE AS expiry_date,
               v_today_date,
               v_validity_months,
               0 AS month_diff,
               0 AS late_months,
               0::NUMERIC AS late_fee,
               v_base_fee AS base_fee,
               v_base_fee AS total_fee,
               'NO LATE FEE FOR NON-RENEWAL TYPE' AS status;
        RETURN;
    END IF;

    -- 🔹 4️⃣ Renewal Calculation
    v_month_diff := (EXTRACT(YEAR FROM AGE(v_today_date, v_expiry_date)) * 12)
                    + EXTRACT(MONTH FROM AGE(v_today_date, v_expiry_date));

    v_late_months := v_month_diff - v_validity_months;

    -- Case 1: Within validity → no late fee
    IF v_late_months <= 0 THEN
        v_late_fee := 0;
        v_total_fee := v_base_fee;
```

```
        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
             v_late_months, v_late_fee, v_base_fee, v_total_fee,
             'NORMAL RENEWAL - NO LATE FEE';

    -- Case 2: Expired more than validity → treat as new
    ELSIF v_late_months > v_validity_months THEN
        SELECT f.fees
        INTO v_fresh_fee
        FROM tnelb_fees f
        WHERE f.cert_licence_id = p_licence_id
          AND f.fees_type = 'N'
          AND f.start_date <= CURRENT_DATE
        ORDER BY f.start_date DESC
        LIMIT 1;

        v_late_fee := 0;
        v_total_fee := v_fresh_fee;

        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
             v_late_months, v_late_fee, v_fresh_fee AS base_fee, v_total_fee,
             'EXPIRED > 12 MONTHS - NEW FEE APPLICABLE';

    -- Case 3: Expired within 12 months → late fee applies
    ELSE
        SELECT f.fees
        INTO v_late_fee_per_month
        FROM tnelb_fees f
        WHERE f.cert_licence_id = p_licence_id
          AND f.fees_type = 'L'
          AND f.start_date <= CURRENT_DATE
        ORDER BY f.start_date DESC
        LIMIT 1;

        v_late_fee := v_late_months * v_late_fee_per_month;
        v_total_fee := v_base_fee + v_late_fee;

        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
             v_late_months, v_late_fee, v_base_fee, v_total_fee,
             'LATE FEE APPLICABLE (AFTER EXPIRY)';
    END IF;
END;
$BODY$;

ALTER FUNCTION public.calc_late_fee_dynamic(character varying, integer, character
varying)
```

```
    OWNER TO postgres;
```

2.)

```
-- FUNCTION: public.calc_late_fee_static()

-- DROP FUNCTION IF EXISTS public.calc_late_fee_static();

CREATE OR REPLACE FUNCTION public.calc_late_fee_static(
        )
    RETURNS TABLE(expiry_date date, today_date date, validity_months integer,
month_diff integer, late_months integer, late_fee numeric, base_fee numeric, total_fee
numeric, status text)
    LANGUAGE 'plpgsql'
    COST 100
    VOLATILE PARALLEL UNSAFE
    ROWS 1000

AS $BODY$
DECLARE
    -- 🧩 Static values for testing
    v_today_date DATE := '2025-7-11';     -- current date
    v_expiry_date DATE := '2025-11-11';    -- expiry date of licence
    v_validity_months INT := 3;           -- grace period (3 months)
    v_late_fee_per_month NUMERIC := 300;   -- Rs. 300 per month late fee
    v_base_fee NUMERIC := 1500;           -- base renewal fee
    v_fresh_fee NUMERIC := 2500;          -- new application fee

    v_month_diff INT := 0;
    v_late_months INT := 0;
    v_late_fee NUMERIC := 0;
    v_total_fee NUMERIC := 0;
BEGIN
    -- 🧮 Calculate months between expiry and today
    v_month_diff := (EXTRACT(YEAR FROM AGE(v_today_date, v_expiry_date)) * 12)
```

```
            + EXTRACT(MONTH FROM AGE(v_today_date, v_expiry_date));

    -- 🧮 Calculate how many months late (beyond validity)
    v_late_months := v_month_diff - v_validity_months;

    -- 🔹 Case 1: Within validity → no late fee
    IF v_late_months <= 0 THEN
        v_late_fee := 0;
        v_total_fee := v_base_fee;
        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
               v_late_months, v_late_fee, v_base_fee, v_total_fee,
               'NORMAL RENEWAL - NO LATE FEE';

    -- 🔹 Case 2: Expired > 12 months → treat as new
    ELSIF v_month_diff > 12 THEN
        v_late_fee := 0;
        v_total_fee := v_fresh_fee;
        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
               v_late_months, v_late_fee, v_fresh_fee AS base_fee, v_total_fee,
               'EXPIRED > 12 MONTHS - NEW FEE APPLICABLE';

    -- 🔹 Case 3: Expired within 12 months → late fee applies
    ELSE
        v_late_fee := v_late_months * v_late_fee_per_month;
        v_total_fee := v_base_fee + v_late_fee;
        RETURN QUERY
        SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
               v_late_months, v_late_fee, v_base_fee, v_total_fee,
               'LATE FEE APPLICABLE (AFTER EXPIRY)';
    END IF;
END;
$BODY$;

ALTER FUNCTION public.calc_late_fee_static()
    OWNER TO postgres;
```

```
DO $$
BEGIN
   -- RAISE NOTICE '--- Starting Fee Calculation Test ---';

        SELECT * FROM calc_late_fee_dynamic('R', 1, 'LCS000001');

   -- RAISE NOTICE '--- Calculation Completed ---';
END $$ LANGUAGE plpgsql;


select * from mst_fees_validity




-- FUNCTION: public.calc_fees(character varying, integer, character varying)

-- DROP FUNCTION IF EXISTS public.calc_fees(character varying, integer, character
varying);

CREATE OR REPLACE FUNCTION public.calc_fees(
        p_form_type character varying,
        p_licence_id integer,
        p_issued_licence_no character varying DEFAULT NULL::character varying)
   RETURNS TABLE(expiry_date date, today_date date, validity_months integer,
month_diff integer, late_months integer, late_fee numeric, base_fee numeric, total_fee
numeric, fee_type character varying, status text)
   LANGUAGE 'plpgsql'
   COST 100
   VOLATILE PARALLEL UNSAFE
   ROWS 1000

AS $BODY$
DECLARE
   v_today_date DATE := CURRENT_DATE;
   v_expiry_date DATE := NULL;
   v_validity_months INT := 0;
   v_late_fee_per_month NUMERIC := 0;
   v_base_fee NUMERIC := 0;
   v_fresh_fee NUMERIC := 0;
   v_month_diff INT := 0;
   v_late_months INT := 0;
   v_late_fee NUMERIC := 0;
   v_total_fee NUMERIC := 0;
BEGIN
```

```
        SELECT f.sample_date INTO v_today_date
                FROM tnelb_license f
                WHERE f.license_number = 'LCS000001'
                ORDER BY f.id DESC;

        RAISE NOTICE 'Current Date: %', v_today_date;


  -- 🔹 1️⃣ Only fetch expiry when Renewal (R)
  IF p_form_type = 'R' THEN
    IF p_issued_licence_no IS NULL THEN
      RAISE EXCEPTION 'issued_licence_no must be provided for Renewal
(form_type = R)';
    END IF;

    SELECT expires_at
    INTO v_expiry_date
    FROM tnelb_license
    WHERE license_number = p_issued_licence_no
    LIMIT 1;

    IF v_expiry_date IS NULL THEN
      RAISE EXCEPTION 'No record found for issued_licence_no: %',
p_issued_licence_no;
    END IF;
  END IF;

  -- 🔹 2️⃣ Get fee info from tnelb_fees
  SELECT
        f.fees
      INTO v_base_fee
      FROM tnelb_fees f
      WHERE f.cert_licence_id = p_licence_id
        AND f.fees_type = p_form_type
        AND f.start_date <= CURRENT_DATE
      ORDER BY f.start_date DESC
      LIMIT 1;




  IF NOT FOUND THEN
    RAISE NOTICE 'No fee configuration found for licence_id % and form_type %',
p_licence_id, p_form_type;
    RETURN;
  END IF;
```

```
    -- 🧮3️⃣For non-renewal (like F or N)
    IF p_form_type <> 'R' THEN
            RETURN QUERY
            SELECT NULL::DATE AS expiry_date,
                v_today_date,
                v_validity_months,
                0 AS month_diff,
                0 AS late_months,
                0::NUMERIC AS late_fee,      -- ✅ cast to numeric
                v_base_fee AS base_fee,
                v_base_fee AS total_fee,
                p_form_type AS fee_type,
                'NO LATE FEE FOR NON-RENEWAL TYPE' AS status;
            RETURN;
        END IF;

    -- 🔹4️⃣Renewal Calculation
    v_month_diff := EXTRACT(MONTH FROM AGE(v_today_date, v_expiry_date));
    v_late_months := v_month_diff - v_validity_months;

    IF v_late_months <= 0 THEN
       v_late_fee := 0;
       v_total_fee := v_base_fee;
       RETURN QUERY
       SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
            v_late_months, v_late_fee, v_base_fee, v_total_fee,
            p_form_type, 'NORMAL RENEWAL - NO LATE FEE';
    ELSE
                  SELECT
                     f.fees
                  INTO v_late_fee_per_month
                  FROM tnelb_fees f
                  WHERE f.cert_licence_id = p_licence_id
                    AND f.fees_type = 'L'
                    AND f.start_date <= CURRENT_DATE
                  ORDER BY f.start_date DESC
                  LIMIT 1;


       v_late_fee := v_late_months * v_late_fee_per_month;
       v_total_fee := v_base_fee + v_late_fee;
       RETURN QUERY
       SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
            v_late_months, v_late_fee, v_base_fee, v_total_fee,
            p_form_type, 'LATE FEE APPLICABLE (AFTER EXPIRY)';
    END IF;
END;
$BODY$;
```

```
ALTER FUNCTION public.calc_fees(character varying, integer, character varying)
    OWNER TO postgres;
```

```
DECLARE
   v_today_date DATE := NULL;
   v_expiry_date DATE;
   v_validity_months INT := 0;
   v_base_fee NUMERIC := 0;
   v_fresh_fee NUMERIC := 0;
   v_late_fee_per_month NUMERIC := 0;
   v_late_fee NUMERIC := 0;
   v_total_fee NUMERIC := 0;
   v_month_diff INT := 0;
   v_late_months INT := 0;
   v_renewal_start_date DATE;
BEGIN

        SELECT sample_date
      INTO v_today_date
      FROM tnelb_license
      WHERE license_number = 'LCS000001'
      LIMIT 1;

                  RAISE NOTICE 'today: %', v_today_date;
   ---------------------------------------------------------------
   --1️⃣ Get expiry date for renewal case
   ---------------------------------------------------------------
   -- IF p_form_type = 'R' THEN

   -- END IF;

   ---------------------------------------------------------------
   --2️⃣ Fetch base fee + validity from tnelb_fees
   ---------------------------------------------------------------
   SELECT f.fees
   INTO v_base_fee
   FROM tnelb_fees f
   WHERE f.cert_licence_id = p_licence_id
     AND f.fees_type = p_form_type
     AND f.start_date <= CURRENT_DATE
   ORDER BY f.start_date DESC
   LIMIT 1;


   IF v_base_fee IS NULL THEN
      RAISE EXCEPTION 'No fee found for licence_id % and form_type %',
p_licence_id, p_form_type;
   END IF;

   ---------------------------------------------------------------
```

```
   --3️⃣ If not a renewal (like Fresh/New)
   ---------------------------------------------------------------
   IF p_form_type <> 'R' THEN
      RETURN QUERY
      SELECT NULL::DATE, v_today_date, v_validity_months, 0, 0,
           0::NUMERIC, v_base_fee, v_base_fee,
           p_form_type, 'NON-RENEWAL TYPE - BASE FEE ONLY';
      RETURN;
        ELSE
                SELECT expires_at INTO v_expiry_date
      FROM tnelb_license
      WHERE license_number = p_issued_licence_no
      LIMIT 1;

      IF v_expiry_date IS NULL THEN
         RAISE EXCEPTION 'No record found for licence_no: %', p_issued_licence_no;
      END IF;

                SELECT f.validity
                INTO v_validity_months
                FROM mst_fees_validity f
                WHERE f.licence_id = p_licence_id
                  AND f.form_type = 'L'
                  AND f.validity_start_date <= CURRENT_DATE
                ORDER BY f.validity_start_date DESC
                LIMIT 1;

                RAISE NOTICE 'validity: %', v_validity_months;

                IF v_expiry_date >= v_today_date THEN
      RAISE NOTICE 'Licence NOT expired (Expiry: %, Today: %)', v_expiry_date,
v_today_date;

      -- Dynamic renewal start date (based on validity months)
      v_renewal_start_date := (v_expiry_date - (v_validity_months || '
months')::interval)::date;
      RAISE NOTICE 'Renewal allowed from: % (validity months = %)',
v_renewal_start_date, v_validity_months;

                ---------------------------------------------------------------
            --4️⃣ Renewal Logic (Not expired yet)
            ---------------------------------------------------------------
      IF v_today_date < v_renewal_start_date THEN
                -- Too early to renew
                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
                     0::NUMERIC, v_base_fee, 0::NUMERIC,
```

```
                    p_form_type, 'TOO EARLY FOR RENEWAL';
                RETURN;
            ELSE

                -- Within renewal window (before expiry)
                v_month_diff := (EXTRACT(YEAR FROM AGE(v_expiry_date,
v_today_date)) * 12)
                        + EXTRACT(MONTH FROM AGE(v_expiry_date, v_today_date));

                v_late_months := v_validity_months - v_month_diff;
                IF v_late_months < 1 THEN
                   v_late_months := 1;
                END IF;

                -- Get late fee per month
                SELECT f.fees INTO v_late_fee_per_month
                FROM tnelb_fees f
                WHERE f.cert_licence_id = p_licence_id
                  AND f.fees_type = 'L'
                  AND f.start_date <= CURRENT_DATE
                ORDER BY f.start_date DESC
                LIMIT 1;

                v_late_fee := v_late_months * v_late_fee_per_month;
                v_total_fee := v_base_fee + v_late_fee;

                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
                    v_late_months, v_late_fee, v_base_fee, v_total_fee,
                    p_form_type, FORMAT('WITHIN RENEWAL PERIOD - LATE FEE FOR
%s MONTH(S)', v_late_months);
                RETURN;
            END IF;
        END IF;


            ----------------------------------------------------------------
        --5️⃣ Licence Expired
        ----------------------------------------------------------------
        IF v_today_date > v_expiry_date THEN
            -- Expired within 1 year → Fresh fee
            IF v_today_date <= (v_expiry_date + INTERVAL '1 year') THEN
                SELECT f.fees INTO v_fresh_fee
                FROM tnelb_fees f
                WHERE f.cert_licence_id = p_licence_id
                  AND f.fees_type = 'N'
                  AND f.start_date <= CURRENT_DATE
                ORDER BY f.start_date DESC
```

```
                LIMIT 1;

                v_total_fee := v_fresh_fee;

                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
                    0::NUMERIC, v_fresh_fee, v_total_fee,
                    p_form_type, 'EXPIRED - WITHIN 1 YEAR (FRESH FEE
APPLICABLE)';
            ELSE
                -- Expired more than 1 year
                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
                    0::NUMERIC, 0::NUMERIC, 0::NUMERIC,
                    p_form_type, 'EXPIRED BEYOND 1 YEAR - NO FEES APPLICABLE';
            END IF;
        END IF;
    END IF;

END;
```

```
DECLARE
   v_today_date DATE := NULL;
   v_expiry_date DATE;
   v_validity_months INT := 0;
   v_base_fee NUMERIC := 0;
   v_fresh_fee NUMERIC := 0;
   v_late_fee_per_month NUMERIC := 0;
   v_late_fee NUMERIC := 0;
   v_total_fee NUMERIC := 0;
   v_month_diff INT := 0;
   v_late_months INT := 0;
   v_renewal_start_date DATE;
BEGIN

        SELECT sample_date
      INTO v_today_date
      FROM tnelb_license
      WHERE license_number = 'LCS000001'
      LIMIT 1;

                  RAISE NOTICE 'today: %', v_today_date;
   ---------------------------------------------------------------
   --1️⃣ Get expiry date for renewal case
   ---------------------------------------------------------------
   -- IF p_form_type = 'R' THEN

   -- END IF;

   ---------------------------------------------------------------
   --2️⃣ Fetch base fee + validity from tnelb_fees
   ---------------------------------------------------------------
   SELECT f.fees
   INTO v_base_fee
   FROM tnelb_fees f
   WHERE f.cert_licence_id = p_licence_id
     AND f.fees_type = p_form_type
     AND f.start_date <= CURRENT_DATE
   ORDER BY f.start_date DESC
   LIMIT 1;


   IF v_base_fee IS NULL THEN
      RAISE EXCEPTION 'No fee found for licence_id % and form_type %',
p_licence_id, p_form_type;
   END IF;

   ---------------------------------------------------------------
```

```
   --3️⃣ If not a renewal (like Fresh/New)
   ---------------------------------------------------------------
   IF p_form_type <> 'R' THEN
      RETURN QUERY
      SELECT NULL::DATE, v_today_date, v_validity_months, 0, 0,
           0::NUMERIC, v_base_fee, v_base_fee,
           p_form_type, 'NON-RENEWAL TYPE - BASE FEE ONLY';
      RETURN;
        ELSE
                SELECT expires_at INTO v_expiry_date
      FROM tnelb_license
      WHERE license_number = p_issued_licence_no
      LIMIT 1;

      IF v_expiry_date IS NULL THEN
         RAISE EXCEPTION 'No record found for licence_no: %', p_issued_licence_no;
      END IF;

                SELECT f.validity
                INTO v_validity_months
                FROM mst_fees_validity f
                WHERE f.licence_id = p_licence_id
                  AND f.form_type = 'L'
                  AND f.validity_start_date <= CURRENT_DATE
                ORDER BY f.validity_start_date DESC
                LIMIT 1;

                RAISE NOTICE 'validity: %', v_validity_months;

                IF v_expiry_date >= v_today_date THEN
      RAISE NOTICE 'Licence NOT expired (Expiry: %, Today: %)', v_expiry_date,
v_today_date;

      -- Dynamic renewal start date (based on validity months)
      v_renewal_start_date := (v_expiry_date - (v_validity_months || '
months')::interval)::date;
      RAISE NOTICE 'Renewal allowed from: % (validity months = %)',
v_renewal_start_date, v_validity_months;

                ---------------------------------------------------------------
            --4️⃣ Renewal Logic (Not expired yet)
            ---------------------------------------------------------------
      IF v_today_date < v_renewal_start_date THEN
                -- Too early to renew
                              v_total_fee := v_base_fee;
                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
```

```
                    0::NUMERIC, v_base_fee,v_total_fee,p_form_type,
                     'TOO EARLY FOR RENEWAL';
                RETURN;
            ELSE

                -- Within renewal window (before expiry)
                v_month_diff := (EXTRACT(YEAR FROM AGE(v_expiry_date,
v_today_date)) * 12)
                        + EXTRACT(MONTH FROM AGE(v_expiry_date, v_today_date));

                v_late_months := v_validity_months - v_month_diff;
                IF v_late_months < 1 THEN
                   v_late_months := 1;
                END IF;

                -- Get late fee per month
                SELECT f.fees INTO v_late_fee_per_month
                FROM tnelb_fees f
                WHERE f.cert_licence_id = p_licence_id
                  AND f.fees_type = 'L'
                  AND f.start_date <= CURRENT_DATE
                ORDER BY f.start_date DESC
                LIMIT 1;

                v_late_fee := v_late_months * v_late_fee_per_month;
                v_total_fee := v_base_fee + v_late_fee;

                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, v_month_diff,
                     v_late_months, v_late_fee, v_base_fee, v_total_fee,
                     p_form_type, FORMAT('WITHIN RENEWAL PERIOD - LATE FEE FOR
%s MONTH(S)', v_late_months);
                RETURN;
            END IF;
        END IF;


             ----------------------------------------------------------------
        --5️⃣ Licence Expired
        ----------------------------------------------------------------
        IF v_today_date > v_expiry_date THEN
           -- Expired within 1 year → Fresh fee
           IF v_today_date <= (v_expiry_date + INTERVAL '1 year') THEN
              SELECT f.fees INTO v_fresh_fee
              FROM tnelb_fees f
              WHERE f.cert_licence_id = p_licence_id
                AND f.fees_type = 'N'
                AND f.start_date <= CURRENT_DATE
```

```
                ORDER BY f.start_date DESC
                LIMIT 1;

                v_total_fee := v_fresh_fee;

                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
                    0::NUMERIC, v_fresh_fee, v_total_fee,
                    p_form_type, 'EXPIRED - WITHIN 1 YEAR (FRESH FEE
APPLICABLE)';
            ELSE
                -- Expired more than 1 year
                RETURN QUERY
                SELECT v_expiry_date, v_today_date, v_validity_months, NULL::INT,
NULL::INT,
                    0::NUMERIC, 0::NUMERIC, 0::NUMERIC,
                    p_form_type, 'EXPIRED BEYOND 1 YEAR - NO FEES APPLICABLE';
            END IF;
        END IF;
    END IF;

END;
```

Get Payment 👋

```
public function getPaymentDetails(Request $request){
    try {


        $licence_code = $request->licence_code;
        $issued_licence = $request->issued_licence;
        $appl_type = $request->appl_type;

        $licence = MstLicence::where('cert_licence_code',$licence_code)->first();

        if ($appl_type === 'R') {
            $paymentDetails = DB::select("
            SELECT * FROM calc_fees(:appl_type, :licence_id, :issued_licence)
            ", [
                'appl_type' => $appl_type,
                'licence_id' => $licence->id,
                'issued_licence' => $issued_licence,
            ]);

        }
        else {
            $paymentDetails = DB::select("
                SELECT * FROM calc_fees(:appl_type, :licence_id, :issued_licence)
            ", [
                'appl_type' => $appl_type,
                'licence_id' => $licence->id,
                'issued_licence' => null,
            ]);
        }


        if (!empty($paymentDetails)) {
            $fees_details['total_fees'] = $paymentDetails[0]->total_fee;
            $fees_details['lateFees'] = $paymentDetails[0]->late_fee;
            $fees_details['late_months'] = $paymentDetails[0]->late_months;
            $fees_details['basic_fees'] = $paymentDetails[0]->base_fee;
        }


        // $fees_details['certificate_name'] = $fees->licence_name;

        return response()->json([
            'status' => 'success',
            'fees_details' => $fees_details,
        ], 200);
```

```
    } catch (Exception $e) {
         return response()->json([
            'status' => 'error',
            'message' => 'Something went wrong. ' . $e->getMessage(),
        ], 500);
    }
}
```