# BAB III

# METODOLOGI PENELITIAN

## 3.1 Gambaran Umum Objek Penelitian

### 3.1.1 PT XYZ

PT XYZ adalah sebuah perusahaan pembiayaan yang beroperasi di Indonesia, didirikan pada tahun 1990. Pada awalnya, fokus utama perusahaan ini adalah sebagai penyedia layanan pembiayaan untuk mendukung penjualan sepeda motor merek Honda di Indonesia. Seiring berjalannya waktu, PT XYZ mengalami pertumbuhan yang cepat dan memperluas cakupan bisnisnya dengan menawarkan berbagai produk pembiayaan, termasuk kendaraan roda empat, sepeda motor, alat berat, serta pembiayaan konsumen seperti pembiayaan elektronik dan kredit multiguna. Setelah mayoritas sahamnya diakuisisi oleh salah satu Bank terkemuka di Indonesia pada tahun 2006, PT XYZ menjadi bagian dari kelompok perusahaan bank tersebut. Dengan jaringan cabang yang luas di seluruh Indonesia, PT XYZ telah menjadi salah satu perusahaan pembiayaan terbesar di negara ini.

Perusahaan ini juga dikenal dengan program pembiayaan inovatifnya, seperti penggunaan sistem *online* dan digital. PT XYZ tidak hanya menawarkan pembiayaan konvensional, tetapi juga menyediakan solusi pembiayaan syariah sesuai dengan kebutuhan nasabah. Sebagai perusahaan pembiayaan terkemuka, PT XYZ memainkan peran vital dalam perekonomian Indonesia dengan memberikan akses pembiayaan kepada masyarakat dan usaha kecil menengah. Dalam melakukan transaksi, Perusahaan ini membutuhkan data diri para nasabah unutk memvalidasi nasabahnya, diantaranya adalah data KTP, NPWP, dan SIM.

Pada penelitian ini objek penelitian secara spesifik yang diteliti adalah divisi IT Arsitektur PT XYZ. Divisi IT Arsitektur memiliki

tanggung jawab yang penting dalam sebuah perusahaan. Tugas utamanya adalah merancang, mengembangkan, dan memelihara arsitektur teknologi informasi yang sesuai dengan kebutuhan perusahaan. Mereka bertanggung jawab untuk merencanakan infrastruktur IT yang efisien dan skalabel, serta memastikan keamanan dan kepatuhan terhadap regulasi dalam pengembangan sistem dan aplikasi. Manfaat dari divisi IT Arsitektur adalah memastikan bahwa infrastruktur dan aplikasi IT bekerja secara optimal, meningkatkan efisiensi operasional, meminimalkan risiko keamanan, dan mendukung inovasi dan pertumbuhan perusahaan secara keseluruhan. Dengan pemahaman mendalam tentang teknologi dan kebutuhan bisnis, divisi ini berperan penting dalam mengarahkan strategi IT yang dapat mendukung visi dan tujuan perusahaan dalam era digital saat ini.

### 3.1.2 KTP

Kartu Tanda Penduduk (KTP) adalah dokumen identifikasi resmi yang dikeluarkan oleh pemerintah suatu negara kepada warganya. KTP berfungsi sebagai bukti bahwa seseorang adalah penduduk resmi negara tersebut dan memiliki identitas yang sah. Dokumen ini mencakup informasi pribadi seperti nama, tempat dan tanggal lahir, jenis kelamin, alamat, dan nomor identifikasi unik. KTP umumnya diperlukan dalam berbagai transaksi dan kegiatan sehari-hari, termasuk pembukaan rekening bank, pendaftaran sekolah, pemilihan umum, serta sebagai syarat administratif lainnya. Pada umumnya, KTP harus diperbaharui secara berkala sesuai dengan ketentuan yang berlaku di negara tersebut. KTP adalah instrumen yang esensial untuk mengidentifikasi dan melindungi hak serta kewajiban warganegara, serta mendukung kelancaran berbagai aspek kehidupan dalam suatu negara.

### 3.1.3 NPWP

Nomor Pokok Wajib Pajak (NPWP) adalah identifikasi yang diberikan kepada wajib pajak di Indonesia. Setiap individu atau entitas usaha yang memiliki kewajiban untuk membayar pajak di Indonesia diharuskan memiliki NPWP. NPWP digunakan untuk mengidentifikasi wajib pajak dalam sistem perpajakan di Indonesia. Nomor ini digunakan dalam proses pelaporan pajak, pembayaran pajak, dan interaksi dengan Direktorat Jenderal Pajak. NPWP memiliki peran krusial dalam administrasi perpajakan di Indonesia, dan setiap wajib pajak diwajibkan untuk memiliki dan menggunakan NPWP sesuai dengan ketentuan perundang-undangan pajak yang berlaku.

### 3.1.4 SIM

Surat Izin Mengemudi (SIM) adalah dokumen resmi yang dikeluarkan oleh pihak berwenang di suatu negara yang memberikan hak kepada individu untuk mengemudi kendaraan bermotor. SIM menunjukkan bahwa pemiliknya telah memenuhi persyaratan tertentu dan dianggap kompeten untuk mengemudikan kendaraan secara hukum. SIM biasanya dibagi menjadi beberapa jenis berdasarkan kategori kendaraan atau kelasnya, seperti SIM A untuk kendaraan roda empat, SIM B1 untuk kendaraan roda dua, dan lain sebagainya. Surat Izin Mengemudi adalah dokumen penting yang menunjukkan kemampuan seseorang untuk mengemudikan kendaraan secara legal. Penting untuk selalu mengikuti peraturan lalu lintas dan memastikan SIM tetap berlaku serta sesuai dengan persyaratan hukum yang berlaku.

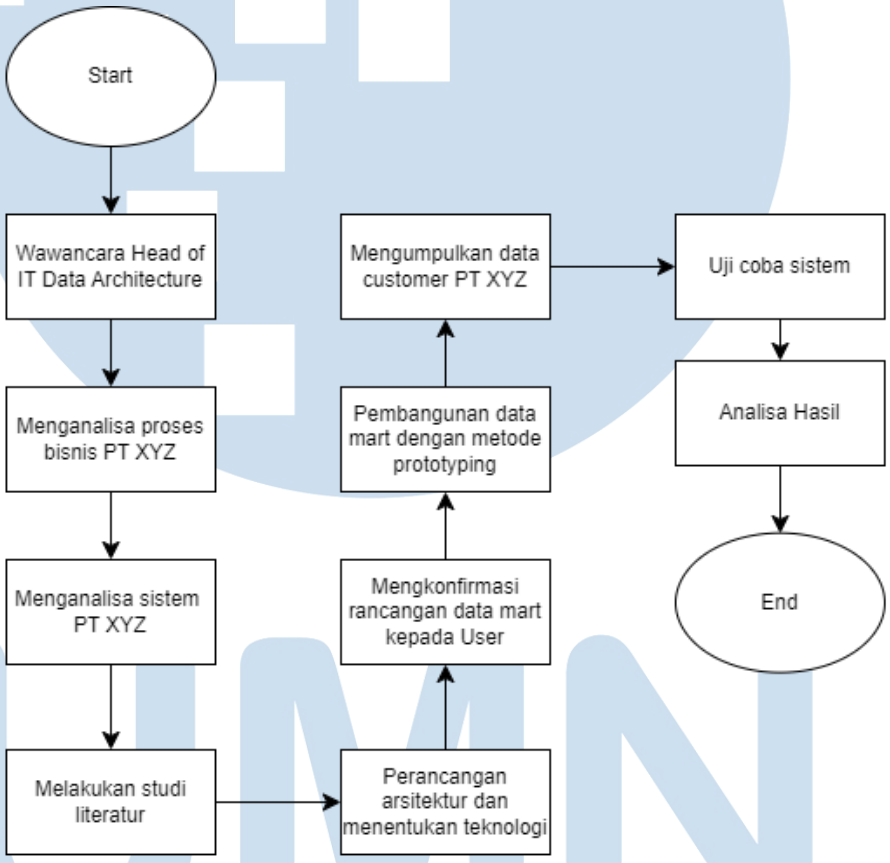
### *3.1.5* JSON *File*

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan oleh komputer. Dibuat berdasarkan bahasa pemrograman JavaScript, JSON tidak tergantung pada bahasa pemrograman tertentu karena menggunakan gaya bahasa yang umum digunakan oleh banyak *programmer*. Hal ini membuat JSON ideal sebagai bahasa pertukaran

data [45]. Pada penelitian ini, JSON *File* berisi informasi seperti tipe kendaraan, jumlah kredit, tenor, hingga *rate* pada *invoice* dari nasabah PT XYZ yang melakukan pengajuan kredit. JSON *File* akan diolah hingga menghasilkan jumlah angsuran perbulan yang harus dibayarkan oleh nasabah.

## 3.2 Metode Penelitian

### 3.2.1 Alur Penelitian




Gambar 3.1 Alur penelitian

Gambar 3.1 merupakan alur penelitian yang dilakukan di dalam penelitian ini. Tahap pertama, yaitu melakukan wawancara mendalam dengan *Head of* IT Data Arsitektur dan tim IT Teknologi Arsitektur di PT XYZ. Wawancara ini menjadi langkah kunci untuk mendapatkan pemahaman yang mendalam mengenai permasalahan yang dihadapi dan kebutuhan yang perlu dipenuhi oleh PT XYZ. Tahap kedua dilanjutkan dengan melakukan analisis terhadap proses bisnis yang berjalan di PT XYZ.

Analisis ini bertujuan untuk memahami proses dan alur perolehan data nasabah yang menjadi fokus penelitian. Selanjutnya, tahap ketiga adalah menganalisis sistem yang digunakan oleh PT XYZ. Melalui analisis ini, dapat diperoleh gambaran mengenai lokasi penyimpanan data nasabah serta proses pengolahan data yang telah ada. Tahap keempat melibatkan studi literatur yang mendalam untuk mencari solusi yang sesuai dengan permasalahan dan kebutuhan yang dihadapi oleh PT XYZ.

Setelah itu, tahap kelima adalah merancang arsitektur sistem yang optimal dan memilih teknologi yang tepat untuk diimplementasikan dalam pembangunan *data mart*. Dalam tahap ini, telah menghasilkan pemilihan teknologi yaitu menggunakan sistem di *cloud.* Tahap keenam dilakukan dengan melakukan konfirmasi kepada pengguna terkait arsitektur sistem yang telah dirancang, untuk memastikan bahwa solusi yang diajukan sesuai dengan harapan dan kebutuhan mereka. Selanjutnya, tahap ketujuh melibatkan uji coba sistem untuk mengevaluasi kinerjanya sesuai dengan tujuan yang telah ditetapkan. Terakhir, tahap kedelapan adalah menganalisis hasil uji coba untuk menilai apakah sistem yang dikembangkan telah memenuhi harapan dan tujuan yang telah ditetapkan sebelumnya. Dengan demikian, alur penelitian yang terstruktur ini diharapkan mampu memberikan solusi yang tepat serta memberikan nilai tambah yang signifikan bagi PT XYZ dalam mengatasi permasalahan dan memenuhi kebutuhan mereka.

### 3.2.2 Metode Pengembangan Sistem

#### *3.2.2.1* Metode Kimball *(Bottom-up)*

Metode Kimball, dikembangkan oleh Ralph Kimball pada 1980-an, adalah metode pengembangan data warehouse dengan pendekatan bottom-up. Metode *bottom-up* adalah pendekatan merancang dan mengimplementasikan dari tingkat yang paling rendah atau detail terlebih dahulu, kemudian secara bertahap membangun struktur yang lebih kompleks. Pendekatan ini dimulai dengan mengumpulkan data mentah dari sumber-sumber yang berbeda,

kemudian mengorganisasikannya ke dalam dimensi dan fakta yang relevan. Dengan pendekatan ini, *data mart* dibangun dari bawah ke atas, diawali dari tingkat terendah hingga tingkat yang lebih tinggi, seperti agregasi data. Pendekatan *bottom-up* ini sering dipilih karena memungkinkan pengembangan *data mart* yang cepat dan adaptasi yang lebih fleksibel terhadap perubahan kebutuhan bisnis. Pada penelitian ini data mart dibangun mulai dari identitas data nasabah yang tidak terstruktur.

Berikut tabel 3.1 menampilkan perbandingan metode bottom up dengan matode top down:

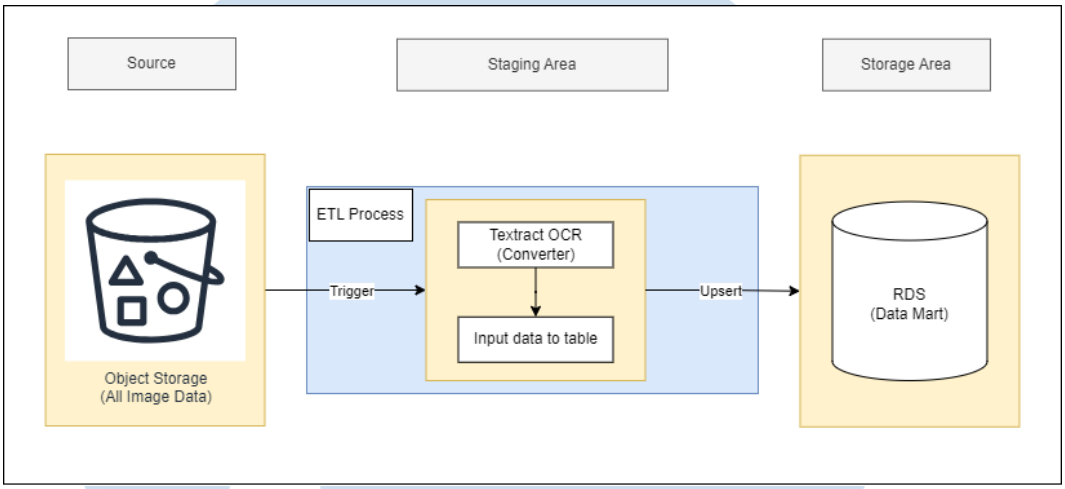
Tabel 3.1 Perbedaan metode

| Aspek | Metode *Bottom-up* | Metode *Top-down* |
|---|---|---|
| Pendekatan | Dimulai dari tingkat data yang paling rendah dan detail, kemudian secara bertahap membangun struktur yang lebih kompleks | Dimulai dengan merancang struktur secara keseluruhan, kemudian menyesuaikan detail dan data yang diperlukan |
| Proses | Data mentah dikumpulkan dari sumber atau bentuk yang berbeda, kemudian diorganisasi ke dalam dimensi dan fakta yang relevan | Struktur data mart dirancang terlebih dahulu berdasarkan kebutuhan bisnis dan penggunaan, kemudian diisi dengan data |
| Durasi Pembangunan | Lebih cepat karena memungkinkan pembangunan dan iterasi yang cepat berdasarkan kebutuhan bisnis yang spesifik | Lebih lambat karena memerlukan perencanaan dan desain yang lebih komprehensif sebelum implementasi sebenarnya |
| Fleksibilitas | Lebih fleksibel dalam mengakomodasi perubahan kebutuhan bisnis karena dapat menyesuaikan struktur dan data yang dibutuhkan secara bertahap | Kurang fleksibel karena perubahan dalam kebutuhan bisnis mungkin memerlukan perubahan besar dalam struktur yang sudah direncanakan sebelumnya |
| Kesesuaian Kebutuhan | Cocok untuk organisasi yang memiliki kebutuhan bisnis yang beragam dan kompleks, serta membutuhkan adaptasi yang cepat terhadap perubahan pasar | Cocok untuk proyek yang memiliki persyaratan bisnis yang jelas dan konsisten, serta memungkinkan perencanaan yang matang dari awal |

Oleh karena itu, berikut merupakan desain arsitektur pembangunan *data mart* :

### 3.3 Desain Arsitektur Data Mart

Pada Gambar 3.2 merupakan desain arsitektur data mart yang akan dibangun




Gambar 3.2 Arsitektur *Data Mart*

Berikut penjelasan arsitektur data mart tersebut :

1. *Source*

   Porses extract dilakukan dengan mengambil sumber data utama yang digunakan dalam penelitian dari *object storage* Amazon S3. Data yang diambil yaitu foto KTP, NPWP, SIM dan file JSON yang berisi informasi mengenai *invoice* nasabah tersebut dan sudah di *zip*. Kriteria data yang diambil yaitu untuk data foto minimal 300dpi**.**

2. *Staging Area*

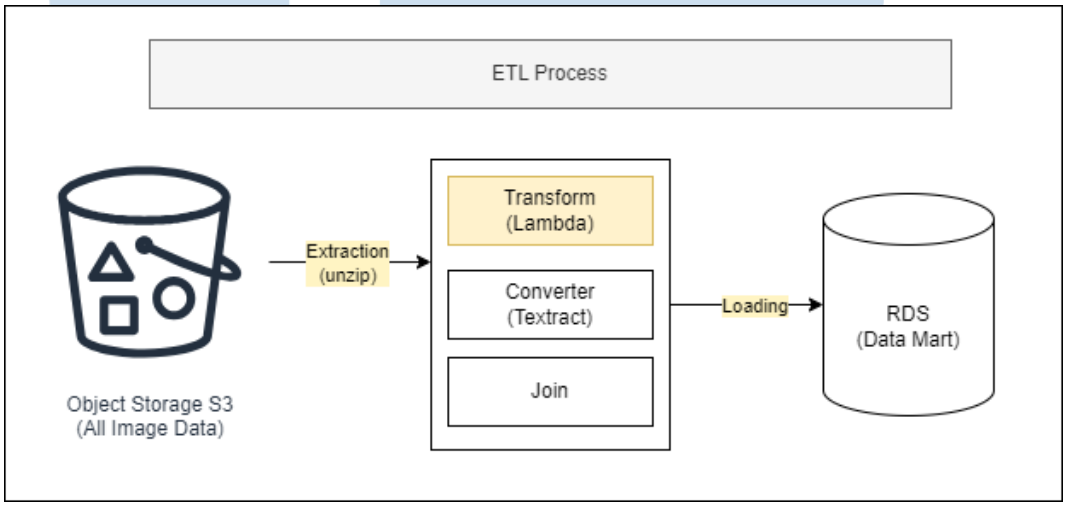
   Pada *staging area* ini dilakukan proses ETL transformasi. ETL dilakukan mulai dari proses *unzip* data kemudian data yang semula tidak terstruktur diubah menjadi terstruktur menggunakan layanan textract. Kemudian hasil dari textract akan dimasukan ke dalam tabel. Pada area ini juga dilakukan perhitungan jumlah angsuran perbulan yang harus dibayarkan nasabah dari *invoice* di file JSON.

3. *Storage Area*

Pada *storage area* ini merupakan proses *Load* dari data yang sudah terstruktur untuk dimasukkan kedalam data mart identitas nasabah yang disimpan dalam *database* AWS RDS. Data mart ini dapat digunakan untuk penunjang data tambahan seperti untuk *Customer Data Management, Customer 360,* atau untuk *reporting*.

## 3.4 ETL *Process*

Gambar 3.3 merupakan Rancang bangun data mart identitas nasabah melalui proses ETL (*Extract, Transform dan Load*).




Gambar 3.3 ETL Proses

Dalam tahapan awal yang dilakukan adalah pengambilan data *zip* dari sumber data yang tersimpan di *bucket* S3 ke dalam area transformasi. Untuk melaksanakan langkah ini, penggunaan *trigger* pada layanan AWS Lambda diimplementasikan. Selanjutnya, pada tahapan kedua, dilakukan ekestraksi data *zip file* dan setelah itu dilakukan proses transformasi menggunakan layanan Textract guna mengubah data yang awalnya tidak terstruktur menjadi terstruktur dan juga melakukan perhitungan jumlah angsuran dari *file* JSON. Setelah proses konversi selesai, data diidentifikasi sesuai dengan isi data-datanya. Terakhir, setelah semua data teridentifikasi, langkah selanjutnya adalah melakukan pemuatan data (*load*) ke dalam data mart yang tersimpan pada layanan RDS (*Relational Database Service*).

Tahapan ini mengindikasikan tahap akhir dari proses pengolahan data, dimana data yang telah terstruktur dan terkonsolidasi siap digunakan untuk keperluan analisis dan pelaporan lebih lanjut.

### *3.4.1 Extract*

Proses Extract yang dilakukan adalah proses mengambil data sumber identitas nasabah berupa zip yang berisi *image file* dan JSON *file* yang diambil dari *object storage* S3.

### 3.4.2 Transformasi

Pada proses ini *file zip* dibuka dan dimasukan ke dalam variabel yang telah dibuat (zip_data), kemudian terjadi proses pemeriksaan setiap file yang ada di dalmnya untuk mencari KTP, SIM, NPWP dan *Invoice*. Jika ditemukan maka terjadi proses penyalinan isinya ke dalam *file* untuk konten. Detail dapat dilihat pada Gambar 3.4,

```
# Membaca isi file zip
with zipfile.ZipFile(io.BytesIO(zip_data)) as zip_ref:
    # Mendapatkan daftar nama file di dalam zip
    file_list = zip_ref.namelist()
    # Proses setiap file di dalam zip
    for file_name in file_list:
        # Baca isi file
        with zip_ref.open(file_name) as file:
            file_content = file.read()
            # Lakukan sesuatu dengan isi file, misalnya, cetak isi file
            try:
                if 'ktp' in file_name:
                    files['ktp'] = file_content
                if 'sim' in file_name:
                    files['sim'] = file_content
                elif 'npwp' in file_name:
                    files['npwp'] = file_content
                elif 'invoice' in file_name:
                    files['invoice'] = file_content
            except UnicodeDecodeError:
                print(f"UnicodeDecodeError occurred while processing {file_name}. Skipping this file.")
```

Gambar 3.4 *unzip file*

Kemudian dilanjutkan dengan *mengimport* 4 *library* ini :

```
import boto3
import io
import zipfile
import json
```

Gambar 3.5 *library*

Pada gambar 3.5, terdapat boto3 yang digunakan untuk berinteraksi dengan layanan AWS seperti *bucket* S3. Io digunakan untuk menyediakan

*interface input/output*. JSON digunakan untuk mengurai data json menjadi objek python.

### 3.4.2.1 Kartu Tanda Penduduk (KTP)

Proses transform objek KTP pada Gambar 3.6 dilakukan sebagai berikut :


Gambar 3.6 Foto KTP

Data yang akan diambil dari KTP ini terdapat pada Gambar 3.7,

```
def extract_ktp(response):
    # 'schema' dari ktp
    key_val = {
        'Domisili': '',
        'Dom': '',
        'NIK': '',
        'Nama': '',
        'Jenis Kelamin': '',
        'Gol. Darah': '',
        'Tempat Lahir': '',
        'Tanggal Lahir': '',
        'Alamat': '',
        'RT/RW':'',
        'Kel/Desa':'',
        'Kecamatan': '',
        'Agama': '',
        'Status Perkawinan': '',
        'Pekerjaan': '',
        'Kewarganegaraan': '',
        'Berlaku Hingga': '',
    }
```

Gambar 3.7 Atribut KTP

Logika utama yang dipakai dalam mengambil data tersebut yaitu dengan memisahkan kunci dan *value* menggunakan " : ". Dapat dilihat di Tabel 3.2,

Tabel 3.2 Logika KTP

```
# Get the first element (key)
SET temp_key TO ITEM(text_list, 1)
# If there are more than 1 elements (key-value pair)
IF LENGTH(text_list) > 1 THEN
  # Get the second element (value) and remove whitespaces
  SET temp_val TO TRIM(ITEM(text_list, 2))
ELSE
  # Print the entire text if there's no colon (no value)
  PRINT(temp_val)
END IF
```

Kemudian pada Tabel 3.3 terdapat alogritma untuk memisahkan tempat lahir dengan tanggal lahir berdasarkan tanda (,) dan (-). Hasil pisahan yang pertama ditandai sebagai tempat lahir, yang kedua ditandai sebagai tanggal lahir.

Tabel 3.3 algoritma tanggal_lahir dan tempat_lahir

```
IF "," AND "-" IN temp_val THEN
  SET birth_info TO SPLIT(temp_val, ",")
  IF LENGTH(birth_info) == 2 THEN
    SET key_val["Tempat Lahir"] TO ITEM(birth_info, 1)
    SET key_val["Tanggal Lahir"] TO ITEM(birth_info, 2)
    CONTINUE
  END IF
END IF
```

Kemudian pada Tabel 3.4 terdapat algoritma untuk mendeteksi bagian ‘Berlaku Hingga’ dilakukan pengecekan jika terdapat simbol (-) dan panjang karakter sama atau lebih dari 10 atau jika berisi kalimat ‘SEUMUR HIDUP’ berarti nilai tersebut milik ‘Berlaku Hingga’.

Tabel 3.4 algoritma berlaku_hingga

```
IF "-" IN temp_val AND LEN(temp_val) >= 10 THEN
  SET key_val["Berlaku Hingga"] = temp_val
ELSE IF temp_val == "SEUMUR HIDUP" THEN
  SET key_val["Berlaku Hingga"] = temp_val
END IF
```

*Output* dapat dilihat di Gambar 3.8

```
Found this KTP data: {'Domisili': 'JAKARTA SELATAN',
'Dom': 'JABODETABEK', 'NIK': '3602758294658003', 'Nama':
'SITI DEWI MARIA', 'Jenis Kelamin': 'PEREMPUAN', 'Gol. Darah': '-',
'Tempat Lahir': '', 'Tanggal Lahir': '', 'Alamat': 'JL. MANGGA 2',
'RT/RW': '002/002', 'Kel/Desa': 'MANGGA 2', 'Kecamatan': 'MANGGA 2',
'Agama': 'ISLAM', 'Status Perkawinan': 'BELUM KAWIN', 'Pekerjaan':
'PELAJAR/MAHASISWA', 'Kewarganegaraan': 'WNI', 'Berlaku Hingga': 'SEUMUR
HIDUP'}
```

Gambar 3.8 *Output* KTP

### 3.4.2.2 Surat Izin Mengemudi (SIM)

Proses *transform* objek SIM pada Gambar 3.9, Dilakukan sebagai berikut :


Gambar 3.9 Foto SIM

Data yang diambil dari SIM ini terdapat pada Gambar 3.10.

```
def extract_sim(response):
    key_val = {
        'No SIM': '',
        'Tipe SIM': '',
        'Nama': '',
        'Tempat lahir': '',
        'Tanggal Lahir': '',
        'Jenis Kelamin': '',
        'Golongan Darah' : '',
        'Alamat': '',
        'Status': '',
        'Daerah': '',
        'Berlaku sampai': '',
    }
```

Gambar 3.10 Atribut SIM

Logika utama yang digunakan untuk mengambil ini yaitu dengan menandai nomor yang berisi nilai yang akan diambil. Dapat dilihat di Tabel 3.5.

Tabel 3.5 Logika SIM

```
DECLARE temp_key, temp_val, splitted
TRY
  # Attempt to split the text by period
  SET splitted TO SPLIT(text, ".")
CATCH SplitError
  # If there's an error, set splitted to an empty list
  SET splitted TO []
END TRY
# Check if there are any parts after the split
IF LENGTH(splitted) > 0 THEN
  # Get the first part (key) and remove whitespaces
  SET temp_key TO TRIM(ITEM(splitted, 1))
  # If there's a second part (value)
  IF LENGTH(splitted) > 1 THEN
    # Get the second part (value) and remove whitespaces
    SET temp_val TO TRIM(ITEM(splitted, 2))
  ELSE
    # Set value to empty string if there's no value after split
    SET temp_val TO ""
  END IF
END IF
# Print the key and value
PRINT(temp_key)
PRINT(temp_val)
```

Kemudian algoritma untuk mengambil nilai ‘No SIM’ ditandai jika pada nilai terdapat simbol (-) dan panjang nilai sama atau lebih dari 16 dan tidak ada simbol (.) dan (,) pada nilai. Algoritma dapat dilihat pada Tabel 3.6.

Tabel 3.6 algoritma mengambil no_sim

| |
|---|
| IF the character '-' is in the text AND the length of text is greater than or equal to 16 AND '.' is not in the text AND ',' is not in the text:<br>Set key_val['No SIM'] to the stripped text<br>Continue to the next iteration |

*Output* dapat dilihat di Gambar 3.11

```
Found this SIM data: {'No SIM': '0123-4567-891011',
'Tipe SIM': 'A', 'Nama': 'SITI DEWI MARIA', 'Tempat
lahir': 'JAKARTA', 'Tanggal Lahir': '17-08-1945', 'Jenis Kelamin':
'WANITA', 'Golongan Darah': 'O', 'Alamat': 'JL SWADAYA 1 RT 14/09 PEJATEN
TIMUR PASAR MINGGU JAKARTA SELATAN', 'Status': 'PELAJAR/MHS', 'Daerah':
'METRO JAYA', 'Berlaku sampai': ''}
```

Gambar 3.11 *Output* SIM

### 3.4.2.3 Nomor Pokok Wajib Pajak (NPWP)

Proses *transform* objek NPWP pada Gambar 3.12, dilakukan sebagai berikut


Gambar 3.12 Foto NPWP

Data yang diambil dari NPWP ini terdapat pada Gambar 3.13,

```
def extract_new_npwp(response):
    key_val = {
        'KPP': '',
        'NPWP': '',
        'Nama': '',
        'NPWP16': '',
        'Alamat': '',
        'Tanggal Terdaftar': '',
    }
```

Gambar 3.13 Atribut NPWP

Logika utama yang digunakan dapat dilihat di Tabel 3.7,

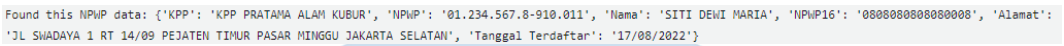
Tabel 3.7 Logika NPWP

```
FUNCTION is_npwp(text):
  parts = Split text by '-' after stripping leading and trailing whitespace
  IF length of parts equals 2:
    first_part = Split parts[0] by '.'
    second_part = Split parts[1] by '.'
    IF length of first_part equals 4 AND length of second_part equals 2 AND all parts in first_part are digits AND all parts in second_part are digits:
      RETURN True
  RETURN False

FUNCTION contains_invalid_keywords(text):
  invalid_keywords = ['djp', 'kpp', 'npwp', 'np', 'vp']
  FOR EACH keyword IN invalid_keywords:
    IF lowercase of keyword is in lowercase of text:
      RETURN True
  RETURN False
```

No NPWP ditandai jika terdapat simbol (.) dan (-) di nilai, untuk nama ditandai jika ditemukan nilai yang tidak mengandung kata ‘djp’, ‘kpp’, ‘npwp’, ‘np’, dan ‘vp’ maka nilai tersebut adalah nama.

Output dapat dilihat di Gambar 3.14




Gambar 3.14 *Output* NPWP

#### 3.4.2.4 JSON *File* (*Invoice*)

Data JSON yang digunakan terdapat pada Gambar 3.15,

```
{
    'TIPE'        : 'Sigra 1.2 R At Mc',
    'HARGA'       : '189700000',
    'DP'          : '67140000',
    'TENOR'       : '11',
    'RATE'        : '0.0199',
    'RESIDU'      : '170000000',
    'ASURANSI'    : '2000000'
}
```

Gambar 3.15 *File JSON*

Ekstrasi file JSON ke untuk mengambil nilai di dalamnya dapat dilihat pada Tabel 3.8.

Tabel 3.8 Ekstrasi JSON

| |
|---|
| # Extract Transaction info<br>invoice_content = Decode 'invoice' file content from UTF-8 and replace single quotes with double quotes<br>invoice_content = Remove carriage return and newline characters from invoice content<br>invoice_dict = Parse invoice_content as JSON<br><br>HARGA = Convert 'HARGA' value from invoice_dict to float, defaulting to 0 if not present<br>DP = Convert 'DP' value from invoice_dict to float, defaulting to 0 if not present<br>RATE = Convert 'RATE' value from invoice_dict to float, defaulting to 0 if not present<br>TENOR = Convert 'TENOR' value from invoice_dict to float, defaulting to 1 if not present<br>RESIDU = Convert 'RESIDU' value from invoice_dict to float, defaulting to 0 if not present<br>ASURANSI = Convert 'ASURANSI' value from invoice_dict to float, defaulting to 0 if not present |

```
angsuran_per_bulan = Calculate installment per month using
hitung_angsuran function with parameters (HARGA, DP, RATE,
TENOR, RESIDU, ASURANSI)
angsuran_dict = Create a dictionary with key 'Angsuran per bulan' and
value angsuran_per_bulan
Print "Angsuran per bulan:" followed by angsuran_per_bulan
```

Perhitungan untuk data JSON ini yaitu terdapat pada Tabel 3.9 berikut:

Tabel 3.9 Perhitungan angsuran_perbulan

```
function hitung_angsuran(harga, dp, rate, tenor, residu, asuransi)
  # Hitung jumlah pinjaman
  jumlah_pinjaman ← harga - dp
  # Hitung bunga per bulan
  bunga ← rate / 12
  # Hitung total angsuran per bulan
  angsuran_per_bulan ← (jumlah_pinjaman + ((jumlah_pinjaman -
residu) * bunga) / tenor) + asuransi
 # Kembalikan nilai angsuran_per_bulan
 return angsuran_per_bulan
```

### 3.4.3 *Loading*

Dapat dilihat pada Tabel 3.10, Pada proses ini dibuat variabel yang menampung hasil transformasi data. Kemudian proses ini memanggil fungsi lambda testskripsi sekaligus mengirimkan hasil transformasi untuk di *load.*

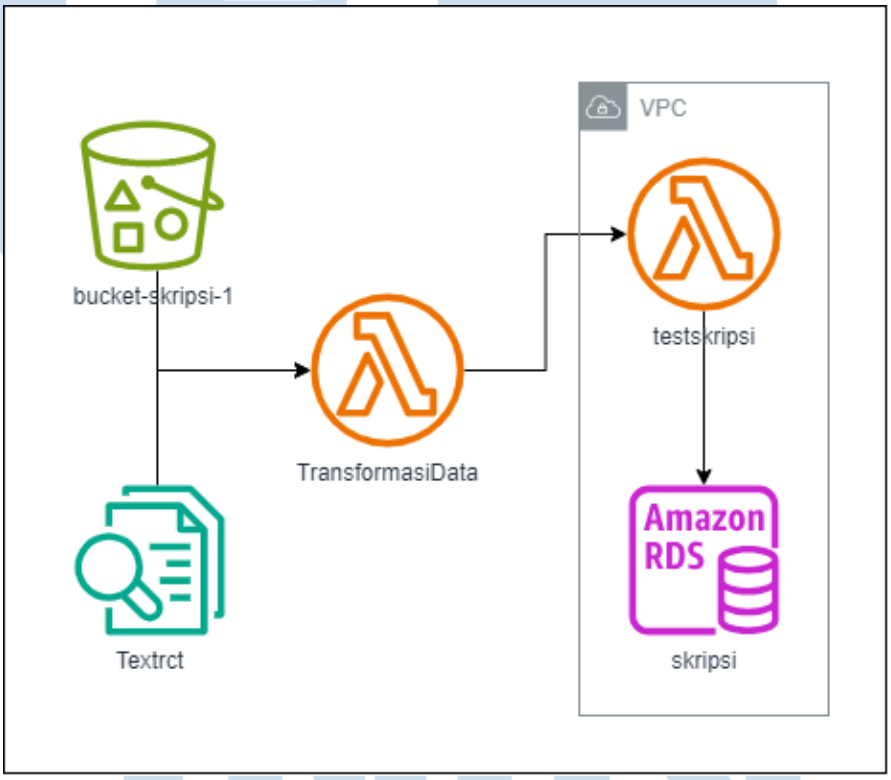
Tabel 3.10 Proses pemanggilan lambda testskripsi untuk load

```
tes_data = {
   "KTP": ktp_dict,
   "SIM": sim_dict,
   "NPWP": npwp_dict,
   "TIPE": invoice_dict,
   "Angsuran": angsuran_dict
}
Print tes_data
```

Invoke Lambda function 'testskripsi' with the following parameters:
- FunctionName: 'testskripsi'
- InvocationType: 'Event'
- Payload: Convert tes_data to JSON format

Kemudian pada lambda testskripsi terjadi proses pemuatan data ke RDS dengan sistem *INSERT* untuk data *invoice* dan *UPSERT* untuk KTP, SIM, NPWP, dan tabel fakta data.

#### 3.2.2.2 Desain Arsitektur Sistem




Gambar 3.16 Arsitektur Sistem

Desain arsitektur sistem yang digambarkan dalam Gambar 3.16 merupakan sebuah arsitektur *serverless* yang memanfaatkan layanan **AWS** (Amazon Web Services) untuk membangun data mart. Arsitektur ini terdiri dari tiga komponen utama, yaitu:

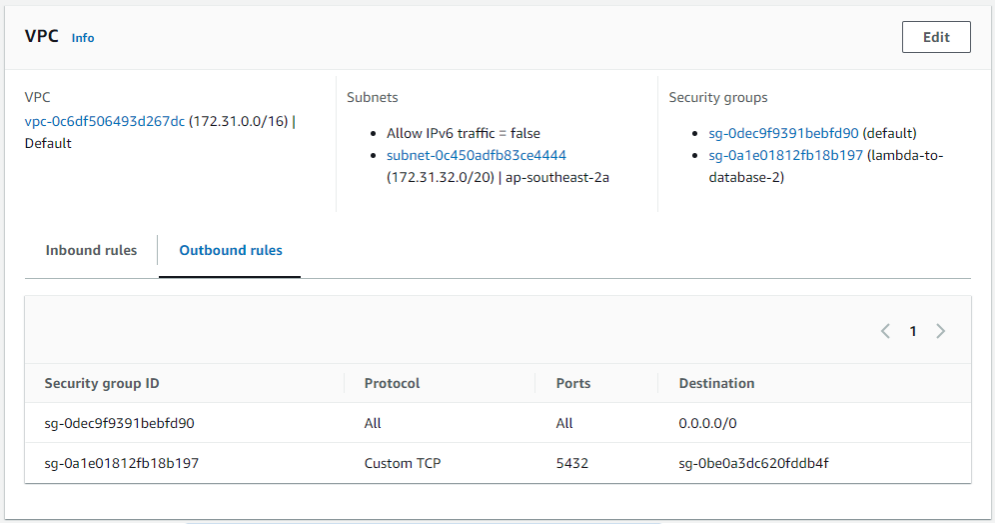
- **Bucket S3** (*Simple Storage Service*): Bucket S3 berfungsi untuk menyimpan data yang akan digunakan dalam data mart. Data tersebut diantaranya foto KTP, NPWP, SIM dan informasi kredit.

Pada penelitian ini, nama *bucket* yang digunakan adalah bucket-skripsi-1 yang telah dikonfigurasi dengan lambda TransformasiData.

- **Lambda** : Lambda adalah fungsi tanpa *server* dimana kita bisa menjalankan kode tanpa harus mengelola server fisik atau infrastruktur komputasi secara langsung. Pada penelitian ini digunakan 2 lambda berbahasa Python.
    - **Lambda 'TransformasiData'** : Lambda ini memiliki peran penting *Extraction* dan *Transform* dalam data mart ini. *Extract* yang dilakukan yaitu dengan menarik data *zip* dari S3 dan melakukan *unzip* file tersebut agar dapat diolah lebih lanjut. *Transform* yang dilakukan memanggil bantuan dari layanan AWS untuk OCR yaitu Textract.
        - **Textract**: Textract adalah layanan yang menggunakan *machine learning* (ML) untuk mengekstrak teks dari dokumen yang dipindai secara otomatis. Pada penelitian ini textract digunakan untuk mengekstrak foto KTP, SIM, dan NPWP. Textract bersama lambda TransformasiData memiliki peran penting dalam proses *Transform* pada ETL.
    - **Lambda 'testskripsi'** : Lambda ini memiliki peran penting dalam proses *Loading* dalam data mart ini. Proses *Loading* dilakukan dengan menggunakan *query* SQL untuk melakukan *UPSERT* untuk tabel dim_ktp, dim_sim, dim_npwp dan data dan *INSERT* untuk table *invoice* ke dalam RDS. Lambda ini berhubungan dengan lambda TransformasiData, hal ini dikarenakan data yang akan di *load* ke RDS merupakan data hasil olahan lambda TransformasiData yang dikirim ke lambda testskripsi. Oleh

karena itu untuk melakukan *execute* dari lambda ini harus dilakukan melalui lambda TransformasiData.

Pada lambda testskripsi, juga digunakan *Virtual Private Cloud* yang berfungsi untuk menghubungkan lambda dengan RDS. Dapat dilihat di Gambar 3.17.




Gambar 3.17 *Virtual Private Cloud*

- **Amazon RDS PostgreSQL**: Amazon RDS PostgreSQL adalah layanan *database* terkelola yang disediakan oleh Amazon Web Services (AWS) untuk menyimpan dan mengelola data terkait dengan data mart nasabah, seperti identitas dan *invoice.* Layanan ini memanfaatkan infrastruktur *cloud* Amazon dan membutuhkan pengaturan *Virtual Private Cloud* (VPC) untuk mengatur akses dan keamanan data. Dalam penelitian ini, Amazon RDS PostgreSQL memiliki sebuah instans yang diberi nama 'skripsi'. Instans ini telah dikonfigurasi dengan tabel dan skema bintang yang berfungsi sebagai data mart. Tabel-tabel ini menyimpan data dalam struktur yang memungkinkan untuk analisis dan pelaporan yang efisien, sesuai dengan kebutuhan penelitian. Dengan menggunakan Amazon RDS PostgreSQL, penelitian dapat mengakses dan mengelola data mart dengan mudah dan aman, tanpa perlu mengelola infrastruktur *database* secara langsung.

Seluruh layanan *cloud* AWS ini dilakukan di AWS *Region* Asia Pacific (Sydney) ap-southeast-2. Ringkasan proses data mart dalam arsitektur ini bekerja sebagai berikut:

1. Data nasabah beserta *invoice* dengan format *zip* diunggah ke *bucket* S3.
2. Perubahan *file* di bucket S3 memicu fungsi Lambda Python TransformasiData.
3. Fungsi lambda TransformasiData memproses data dari bucket S3 dengan memanggil bantuan dari Textract untuk mentransformasi data.
4. Kemudian lambda TransformasiData memerintah lambda testskripsi untuk menjalankan *scriptnya*.
5. Pada fungsi lambda testskripsi berisi *script* untuk menginput hasil proses data tersebut ke RDS.
6. Oleh karena itu hasil pengolahan data akan disimpan di RDS PostgreSQL yang akan menjadi *data mart*.

## 3.3 Teknik Pengumpulan Data

### 3.3.1 Studi Literatur

Penelitian ini mengadopsi teknik pengumpulan data melalui studi literatur sebagai salah satu metode pengumpulan data. Metode ini melibatkan proses sistematis pengumpulan informasi dari berbagai sumber, termasuk riset, jurnal dan literatur yang berkaitan dengan pemanfaatan Data Mart, integrasi teknologi *cloud*, pengelolaan data, dan efisiensi penyimpanan khususnya di sektor *multifinance*. Data dan informasi yang terhimpun dari berbagai literatur ini nantinya akan menjadi dasar referensi yang kuat dan mendalam dalam melakukan penelitian dan membangun sistem ini.

### 3.3.2 Wawancara

Teknik pengumpulan data melalui wawancara ini juga merupakan salah satu teknik penting pada penelitian ini. Hal ini dikarenakan untuk mengidentifikasikan permasalahan dan tujuan penelitian. Wawancara dilakukan bersama dengan manajer IT Data Arsitektur dan tim IT Teknologi Arsitektur pada PT XYZ. Wawancara ini memberikan informasi mengenai salah satu permasalahan yang sedang terjadi di PT XYZ yaitu data foto Identitas yang berada di *object storage* AWS PT XYZ belum diolah secara maksimal. Hal ini mengakibatkan kesulitan menganalisa data. Dengan adanya solusi pembangunan data mart yang dibangun di AWS dan berfokus dengan pengolahan data tidak terstruktur identitas nasabah, maka diharapkan dapat membantu mengolah dan memengoptimalkan data foto identitas.