# 1. API Format Analysis & Calling Pattern Differences

## A. URL and Authentication Methods

**Bigin Systems (ColourCoats & Metalia):**

**Bigin**

**https://www.zohoapis.in/**/bigin/v1/{module_name}?page=1&per_page=200

header

{

"Authorization": "Zoho-oauthtoken your_token" "Content-Type": "application/json"

}

**Rolodex**

**https://api.rolodexcrm.com/user-api/v1/{module}?limit=100&offset=0**

**headers**

{

"Content-Type": "application/json",

"x-rolodex-api-key": "bcdfb94a7e1c331bc201f0059011cfc31e0288af6b57434654f350986174694a"

}

# 1. Data Structure Analysis

| ColourCoats Bigin | Metalia Bigin | Rolodex |
|---|---|---|
| Accounts | Accounts | companies |
| Contacts | Contacts | contacts |
| Deals | Deals | tasks |
| Tasks | Tasks | notes |
| Events | Events | tags |
| Calls | Calls | lists |
| Notes | Notes | custom-fields |
| Products | Products | workspace-users |
| Pipelines | Pipelines | – |
| Associated_Products | Associated_Products | – |
| Attachments | Attachments | – |
| Social | Social | – |
| – | – | – |

Different table names in rolodexcrm and Bigin

## B. ID Systems & Data Types

**Bigin Systems (Both ColourCoats & Metalia):**

- **Structure**: Heavily nested objects with relationship references

**Rolodex:**

- **Structure**: Flatter JSON with direct field access
- **Relationships**: Direct references or separate relationship endpoints

### A. Field Naming Conventions

| Category | ColourCoats Bigin | Metalia Bigin | Rolodex |
|---|---|---|---|
| Company Name | Account_Name | Account_Name | name |
| Contact Name | First_Name, Last_Name, Full_Name | First_Name, Last_Name, Full_Name | first_name, last_name, full_name |
| Phone/Mobile | Phone, Mobile | Mobile, Phone | phone_number |
| Email | Email | Email | email |
| Website | Website | Website | website_url |
| Address Fields | Billing_Street, Billing_City, Billing_State, Billing_Country, Billing_Code | Address_Line_1, Address_Line_2, City, State, PIN_Code | headquarters_location |
| Location Context | City embedded in deals | City (separate field) | location (contacts) |
| Contact Role | Contact_type | Contact_Type, Designation | title |
| Social Media | Instagram | Not present | linkedin_slug, facebook_slug, x_slug, instagram_slug |
| Integration IDs | Rolodex_Company_ID, Rolodex_Contact_ID, Integration_Source | Rolodex_Company_ID, Interakt_Lead_ID, Interakt_Contact_ID, Interakt_Notes, Integration_Source, Last_Sync_Date | Not applicable (source system) |
| Lead Tracking | Lead_Source, Lead_ID | Lead_Source | Not present |
| Pipeline/Opportunity | Stage, Deal_Name | Stage, Deal_Name, Opportunity_Id | Opportunity_Id1 |
| Workspace/Tenant | Not present | Not present | workspace_id |
| Hierarchy | Not present | Not present | manager_id |

**Integration ID Fields Discovered:**

**ColourCoats Bigin:**

- Rolodex_Company_ID: null (prepared for Rolodex company linking)
- Rolodex_Contact_ID: null (prepared for Rolodex contact linking)
- Integration_Source: null (tracks data origin system)

**Metalia Bigin:**

- Rolodex_Company_ID: null (prepared for Rolodex company linking)
- Interakt_Lead_ID: null (WhatsApp/chat platform integration)
- Interakt_Contact_ID: null (WhatsApp contact linking)
- Interakt_Notes: null (chat interaction history)
- Integration_Source: null (tracks data origin system)

**Entity Relationship Patterns**

**Bigin Systems:**

- **Accounts → Contacts**: Nested reference objects
- **Deals → Accounts/Contacts**: Multiple relationship fields
- **Activities**: Linked via What_Id and Who_Id references
- **Tasks/Events**: Related to parent entities via Related_To or Parent_Id

**Rolodex:**

- **Companies ↔ Contacts**: Separate entities with potential linking
- **Lists**: Categorization system for contacts/companies
- **Tasks**: Direct contact associations in arrays
- **Notes**: Workspace-level with rich HTML content

## Challenges issues

## 1. Null Values

- Across **Bigin (ColourCoats & Metalia)** and **Rolodex**, many fields are null or empty.
- Most fields in contacts and companies are either null or not meaningful for mapping.

## 2. Lack of Contact-to-Company Mapping in Rolodex

- In **Rolodex contacts**, there is no company name field in Contacts. Only available fields are: `first_name`, `last_name`, `workspace_id`, and `id`.
- Attempting to use `workspace_id` as a link fails:
    - Multiple workspace users share the same `workspace_id`.
    - Multiple companies share the same `workspace_id`.
- Cannot reliably link a contact to a company in Rolodex.

## 3. Integration IDs Are Null

- In **Bigin (both ColourCoats and Metalia)**, the integration fields (`Rolodex_Company_ID`, `Rolodex_Contact_ID`) are mostly null.
- This prevents mapping between Bigin and Rolodex datasets.
- Without valid IDs, it is not possible to:
    - Identify common companies across datasets.
    - Identify employees working in multiple companies.
    - **Email & Phone Are Unreliable**
- Emails and phone numbers are not always consistent in one company. Cannot be used as a reliable key to link contacts.

## 4. Address Field Differences Between ColourCoats Bigin and Metalia Bigin

- **ColourCoats Bigin:**
  Accounts table contains detailed billing address fields:
    - `Billing_Street`, `Billing_City`, `Billing_State`, `Billing_Code`, `Billing_Country`.
  - These are mostly complete and usable as company addresses.
- **Metalia Bigin:**
  - Accounts table has different address fields:
    - `Address_Line_1`, `Address_Line_2` (mostly null), `City`, `State`, `PIN_Code`.
  - City and state are usable, but street-level address information is often missing.

**Implication:** Fields are inconsistent between the two Bigin systems, requiring normalization to a common format for mapping or analysis.

## 5. Interakt IDs

- `Interakt_Part_ID` exists in **Metalia Bigin** but not consistently in **ColourCoats Bigin**.

## 6. Rolodex Location Field

- Rolodex only has a single `location` field.
- The value often combines **city, state, country** into one string.
- **Challenge:** Requires **normalization** to separate and standardize into usable fields (city, state, country) for mapping or comparison.

## 7. Name initials and inferred title

- Some contacts in ColourCoats Bigin and Metalia Bigin had **initials embedded in the name** that indicate roles (e.g., “Ar. shruti” → “Ar.” = Architect, “Id” = Interior Designer, “Cl” = Client).
  These initials were **not part of the actual name**, I implemented a logic to **filter out or interpret them**.

# Solution Overview

I created 2 dedicated mapper files to standardize and unify heterogeneous data from three different CRM systems into consistent schemas. These mappers serve as the translation layer between disparate source formats and a unified data model.

# Mapper Files Created

## 1. Unified_Companies_mapper.json

Maps company/account data from all three CRM sources to a standardized company schema.

## 2. unified_Personnel_mapper.json

Maps contact/personnel data from all three CRM sources to a standardized personnel schema.

For example

### ColourCoats Bigin Address Handling

Maps structured billing address fields to unified schema:

- `Accounts.Billing_Street` → `address_street`
- `Accounts.Billing_City` → `address_city`
- `Accounts.Billing_State` → `address_state`
- `Accounts.Billing_Code` → `address_postal_code`
- `Accounts.Billing_Country` → `address_country`

### Metalia Bigin Address Handling

Maps different address field structure to same unified fields:

- No street-level mapping (mostly null in source)
- `Accounts.City` → `address_city`
- `Accounts.State` → `address_state`
- `Accounts.PIN_Code` → `address_postal_code`

### Rolodex Location Field Decomposition

Handles compound location strings through string splitting operations:

- `companies.headquarters_location.split(', ')[0]` → `headquarter_city`
- `companies.headquarters_location.split(', ')[1]` → `headquarter_state`

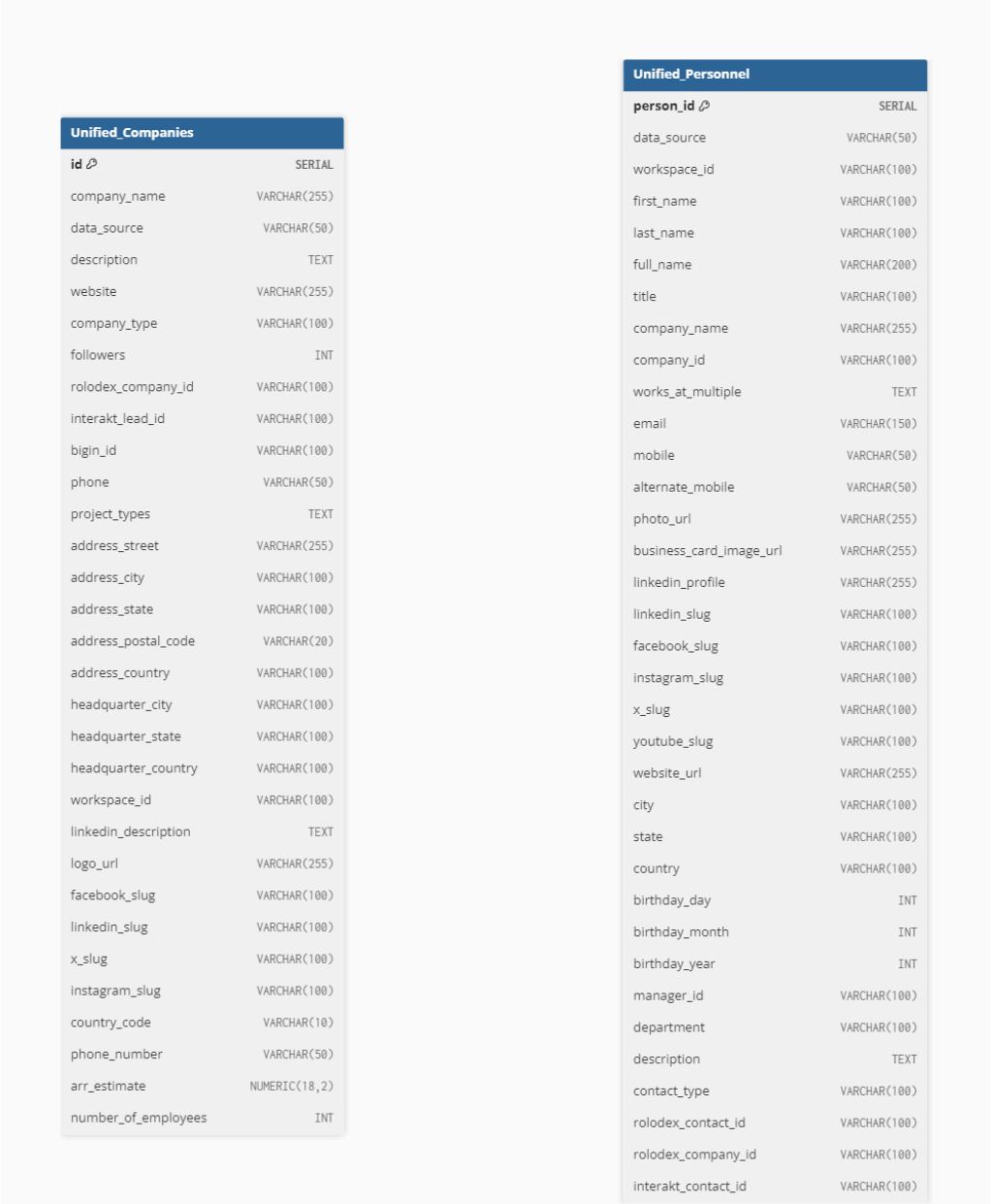
- `companies.headquarters_location.split(', ')[2]` → `headquarter_country`

## Personnel Location Handling

Similar approach for personnel location data:

- Bigin: `Contacts.Mailing_City` → `city`
- Rolodex: `Rolodex_contacts.location.split(', ')[0]` → `city`
- Rolodex: `Rolodex_contacts.location.split(', ')[1]` → `state`
- Rolodex: `Rolodex_contacts.location.split(', ')[2]` → `country`

## Unified schema/tables For firms and personnel



**Unified_Companies**

| id | SERIAL |
|---|---|
| company_name | VARCHAR(255) |
| data_source | VARCHAR(50) |
| description | TEXT |
| website | VARCHAR(255) |
| company_type | VARCHAR(100) |
| followers | INT |
| rolodex_company_id | VARCHAR(100) |
| interakt_lead_id | VARCHAR(100) |
| bigin_id | VARCHAR(100) |
| phone | VARCHAR(50) |
| project_types | TEXT |
| address_street | VARCHAR(255) |
| address_city | VARCHAR(100) |
| address_state | VARCHAR(100) |
| address_postal_code | VARCHAR(20) |
| address_country | VARCHAR(100) |
| headquarter_city | VARCHAR(100) |
| headquarter_state | VARCHAR(100) |
| headquarter_country | VARCHAR(100) |
| workspace_id | VARCHAR(100) |
| linkedin_description | TEXT |
| logo_url | VARCHAR(255) |
| facebook_slug | VARCHAR(100) |
| linkedin_slug | VARCHAR(100) |
| x_slug | VARCHAR(100) |
| instagram_slug | VARCHAR(100) |
| country_code | VARCHAR(10) |
| phone_number | VARCHAR(50) |
| arr_estimate | NUMERIC(18,2) |
| number_of_employees | INT |

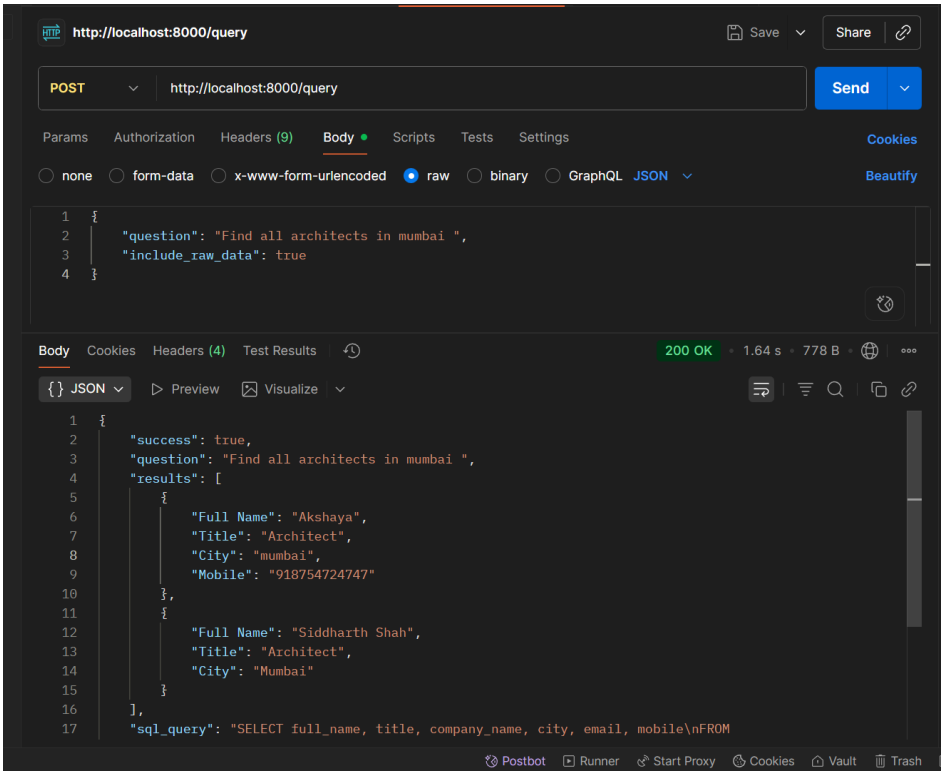
**Unified_Personnel**

| person_id | SERIAL |
|---|---|
| data_source | VARCHAR(50) |
| workspace_id | VARCHAR(100) |
| first_name | VARCHAR(100) |
| last_name | VARCHAR(100) |
| full_name | VARCHAR(200) |
| title | VARCHAR(100) |
| company_name | VARCHAR(255) |
| company_id | VARCHAR(100) |
| works_at_multiple | TEXT |
| email | VARCHAR(150) |
| mobile | VARCHAR(50) |
| alternate_mobile | VARCHAR(50) |
| photo_url | VARCHAR(255) |
| business_card_image_url | VARCHAR(255) |
| linkedin_profile | VARCHAR(255) |
| linkedin_slug | VARCHAR(100) |
| facebook_slug | VARCHAR(100) |
| instagram_slug | VARCHAR(100) |
| x_slug | VARCHAR(100) |
| youtube_slug | VARCHAR(100) |
| website_url | VARCHAR(255) |
| city | VARCHAR(100) |
| state | VARCHAR(100) |
| country | VARCHAR(100) |
| birthday_day | INT |
| birthday_month | INT |
| birthday_year | INT |
| manager_id | VARCHAR(100) |
| department | VARCHAR(100) |
| description | TEXT |
| contact_type | VARCHAR(100) |
| rolodex_contact_id | VARCHAR(100) |
| rolodex_company_id | VARCHAR(100) |
| interakt_contact_id | VARCHAR(100) |

**Task 3.**

Implemented and integrated a FastAPI backend with natural language processing (NLP) capabilities, enabling efficient query handling and scalable API services. The system converts plain English queries into SQL queries using **Google Gemini AI** and returns structured JSON responses. This serves as the backend API that could be integrated with WhatsApp or any chat interface.

Backend: http://localhost:8000/query
Front end : http://localhost:5173/




Postman API endpoint

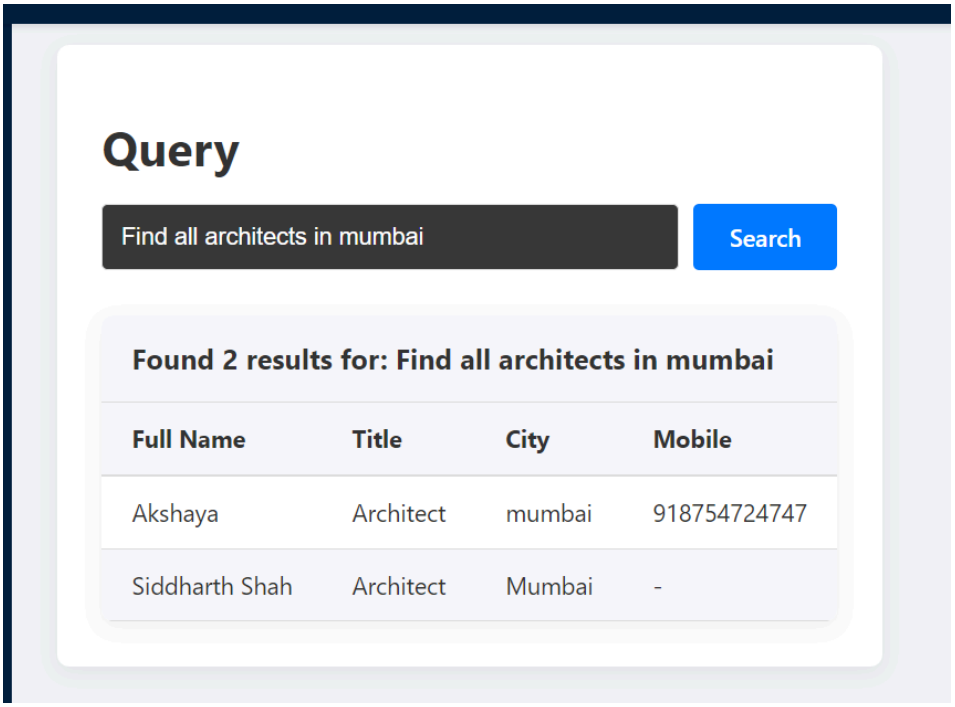



Fig1 Frontend UI 1

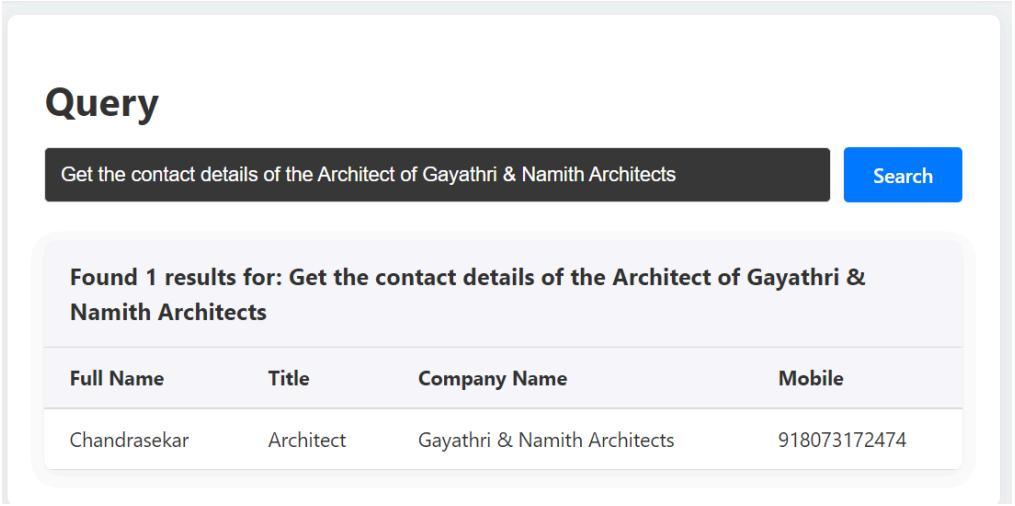



Fig Frontend UI 2