Advanced Programming Final Project

Team: Deepanshi Bansal, Cristian Castillo, Abhijeet Choudhari, Suzanna Newton

Ankota

Ankota is a Boston-based software company focused on creating solutions for organizations that keep older and disabled people living at home. They also support other players in this ecosystem like PACE programs, Area Agencies on Aging (AAAs), Centers for Independent Living (CILs), and more.

Scope :

To build a datamart with ETL capabilities and deliver a report to Ankota on operations as a dashboard.

Software Solutions

Home Care Agency Management Software

Electronic Visit Verification(EVV) Solution

Adult Day Services and Day Habilitation Solutions

Disability LTSS(Long Term Support & Services) and Day Habilitation Solutions

Source: https://www.ankota.com/

Scope of Work

Understand Ankota's work & the home care health industry – 2022 HCP Benchmark Report

Identify key performance indicators

Fetch data from 2 databases: Monday.com & JazzHR

Use this data to build dashboards

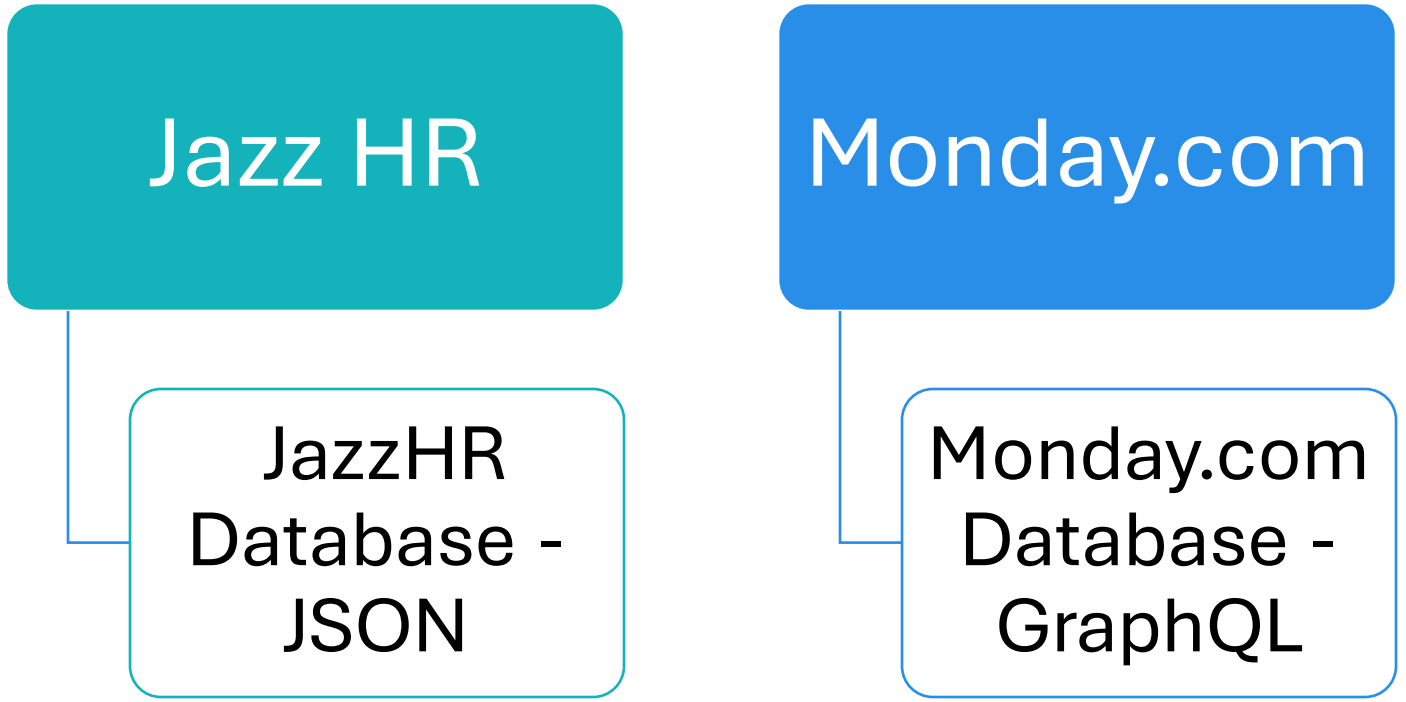
2022-HCP-Benchmarking-Report.pdf

Identification of KPIs

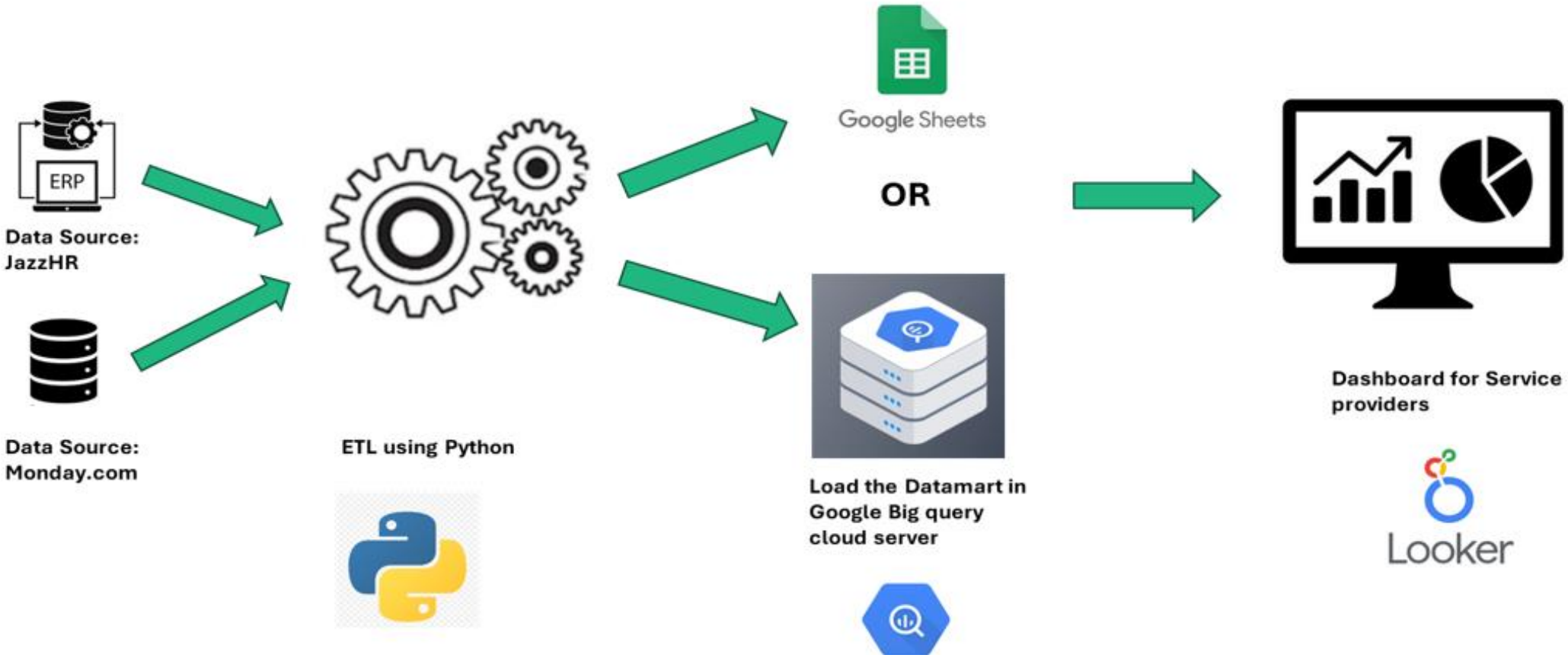
MARKETING & SALES TRACKING

HUMAN RESOURCES TRACKING

API's & Databases Used



Data to Insight Journey



Marketing and Sales Tracking

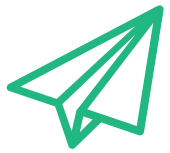
M&S KPIs Identified

Client ID

Status

Marketing Channel (Lead Source)

Inquiry



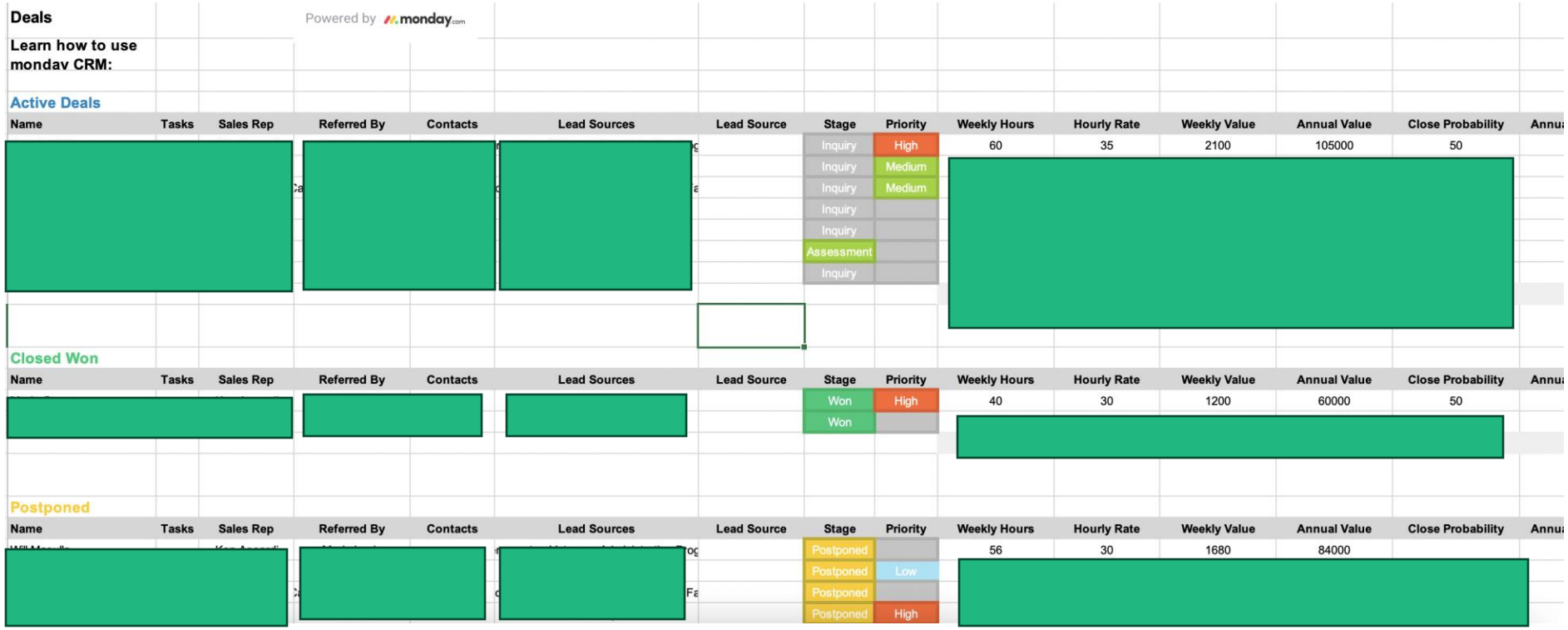
Admission

Assessment

M&S Tracking – Steps Followed

Understanding the Data

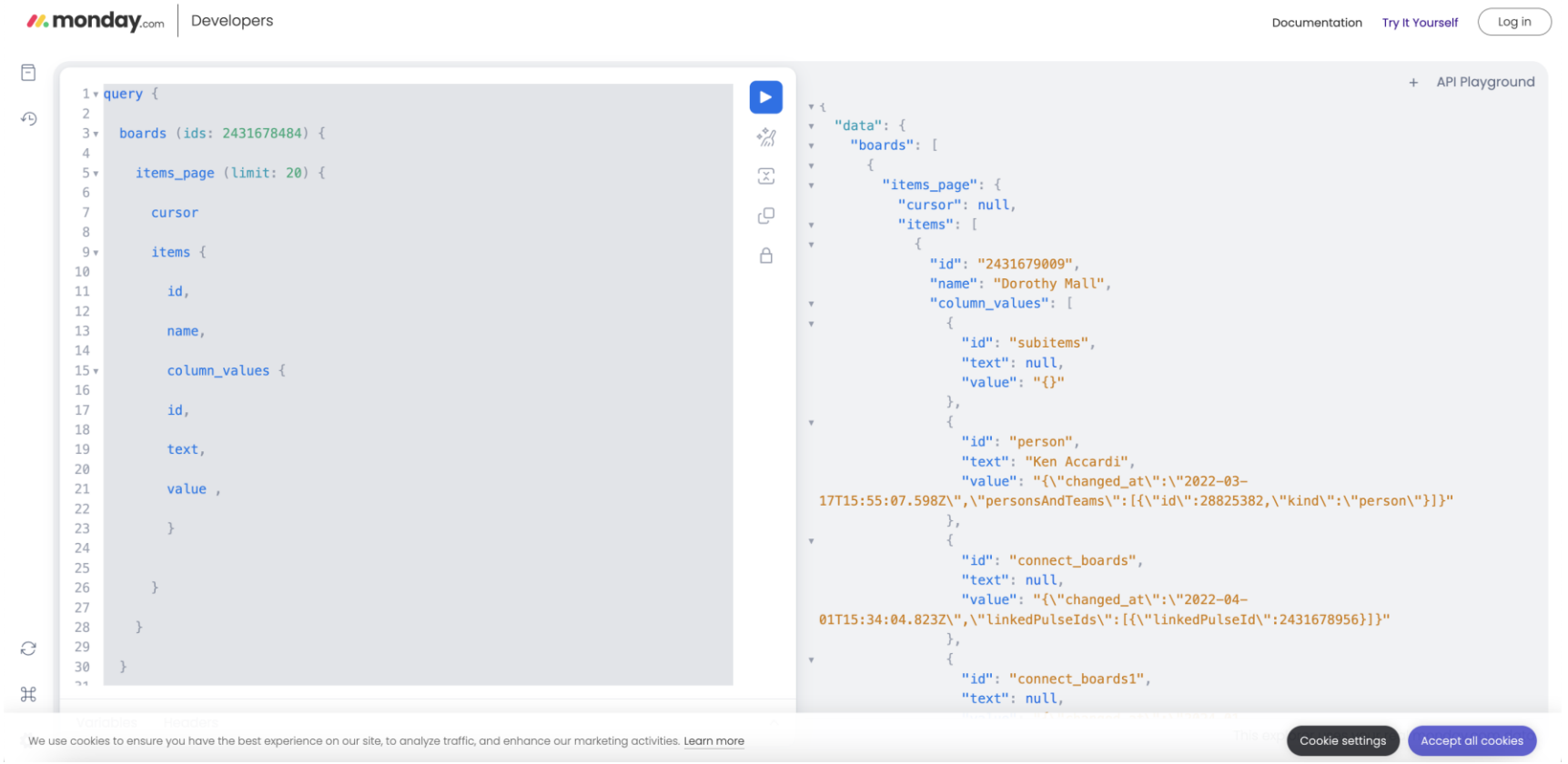
- Data collected via Monday.com was stored in several boards, which could be exported as excel files.



M&S Tracking – Steps Followed

Understanding GraphQL

- Through Monday.com API Playground we got better understanding on GraphQL queries.
- This query fetches information about boards and their columns. It requests the id, name, and columns fields for each board, where the columns field includes the id, title, and type of each column.



M&S Tracking – Steps Followed

Fetching Data on Python

Using the API Key, we extracted the data structure from all the boards on Monday.com and then saved it as a data frame.

Data Structure

In [1]:

```python
import requests
import json
import pandas as pd

# API Key and Headers
api_key ='e[redacted]4f
headers = {[redacted]
api_url = '[redacted]

# Query to fetch boards and their columns
query_boards = """
{
  boards {
    id
    name
    columns {
      id
      title
      type
    }
  }
}
"""

response = requests.post(api_url, json={'query': query_boards}, headers=headers)
boards_data = response.json()['data']['boards']
boards_data
```

Out[1]:

```
[{'id': '5893322376',
  'name': 'New Board',
  'columns': [{'id': 'name', 'title': 'Name', 'type': 'name'},
   {'id': 'person', 'title': 'Person', 'type': 'people'},
   {'id': 'status', 'title': 'Status', 'type': 'status'},
   {'id': 'date4', 'title': 'Date', 'type': 'date'}]},
 {'id': '2431678766',
  'name': 'Subitems of Deals',
  'columns': [{'id': 'name', 'title': 'Name', 'type': 'name'},
   {'id': 'person', 'title': 'Owner', 'type': 'people'},
   {'id': 'status', 'title': 'Status', 'type': 'status'},
   {'id': 'date0', 'title': 'Date', 'type': 'date'}]},
 {'id': '2431678613',
  'name': 'Referred By',
  'columns': [{'id': 'name', 'title': 'Name', 'type': 'name'},
   {'id': 'status', 'title': 'Lead Sources', 'type': 'status'},
   {'id': 'connect_boards3', 'title': 'Contacts', 'type': 'board_relation'},
   {'id': 'mirror1', 'title': 'Deals', 'type': 'mirror'},
   {'id': 'status5', 'title': 'Priority', 'type': 'status'},
   {'id': 'mirror6', 'title': 'Email', 'type': 'mirror'},
   {'id': 'text4', 'title': 'Comments', 'type': 'text'}]},
 {'id': '2431678567',
  'name': 'Contacts',
  'columns': [{'id': 'name', 'title': 'Name', 'type': 'name'},
   {'id': 'link_to___accounts9',
    'title': 'Accounts',
    'type': 'board_relation'},
   {'id': 'link_to___deals', 'title': 'Deals', 'type': 'board_relation'},
   {'id': 'title5', 'title': 'Relationship Type', 'type': 'dropdown'},
   {'id': 'status5', 'title': 'Priority', 'type': 'status'},
   {'id': 'phone', 'title': 'Phone', 'type': 'phone'},
   {'id': 'email', 'title': 'Email', 'type': 'email'},
   {'id': 'long_text4', 'title': 'Comments', 'type': 'long_text'}]},
 {'id': '2431678484',
```

In [2]:

```python
query = '''
{
  boards(limit: 5) {
    name
    id
    columns {
      id
      title
      type
      settings_str
    }
  }
}
'''

data = {'query': query}

r = requests.post(url=api_url, json=data, headers=headers)
response_json = r.json()

# Relevant data for DataFrame
boards_data = response_json['data']['boards']
columns_data = []

for board in boards_data:
    board_id = board['id']
    board_name = board['name']
    for column in board['columns']:
        columns_data.append({
            'Board ID': board_id,
            'Board Name': board_name,
            'Column ID': column['id'],
            'Column Title': column['title'],
            'Column Type': column['type'],
            'Column Settings': column['settings_str']
        })

board_details = pd.DataFrame(columns_data)

board_details
```

Board ID	Board Name	Column ID	Column Title	Column Type	Column Settings
5893322376	New Board	name	Name	name	{}
5893322376	New Board	person	Person	people	{}
5893322376	New Board	status	Status	status	{"done_colors":[1],"labels":{"0":"Postponed","...
5893322376	New Board	date4	Date	date	{}
2431678766	Subitems of Deals	name	Name	name	{}
2431678766	Subitems of Deals	person	Owner	people	{}

M&S Tracking – Steps Followed

Fetching Data on Python

With an understanding of the data structure, we established a data frame for each board by retrieving data from the Monday.com API.

In [3]:

```python
# Initializing a dictionary to hold DataFrames for each board
dfs = {}
#to fetch data from all boards
for board in boards_data:
    board_id = board['id']
    board_name = board.get('name', f"Board_{board_id}")  # Fallback to Board_ID if

    query = '''
    query {
      boards(ids: %s) {
        columns {
          id
          title
        }
        items_page(limit: 20) {
          items {
            id
            name
            column_values {
              id
              text
              value
            }
          }
        }
      }
    }
    ''' % board_id

    response = requests.post(api_url, json={'query': query}, headers=headers)

    if response.status_code == 200:
        board_info = response.json()['data']['boards'][0]
        data = board_info['items_page']['items']
        columns_info = board_info['columns']

        # mapping from column ID to column title
        columns_mapping = {column['id']: column['title'] for column in columns_info

        # Replace unique_columns with column titles instead of IDs
        unique_columns = set(columns_mapping.values())  # Using set to avoid duplic

        # DataFrame columns with 'Item ID' and 'Item Name' always present
        df_columns = ['Item ID', 'Item Name'] + sorted(unique_columns)

        column_values_data = []

        for item in data:
            row_data = {'Item ID': item['id'], 'Item Name': item['name']}
            for column_value in item['column_values']:
                column_title = columns_mapping.get(column_value['id'], 'Unknown Col
                # Attempt to parse 'value' as JSON and extract a meaningful display
                if column_value['text']:
                    row_data[column_title] = column_value['text']
                else:
                    try:
                        value_parsed = json.loads(column_value['value']) if column_
                        if isinstance(value_parsed, dict) and 'text' in value_parse
                            row_data[column_title] = value_parsed['text']
                        else:
                            row_data[column_title] = str(value_parsed)
                    except json.JSONDecodeError:
                        row_data[column_title] = column_value['value']
            column_values_data.append(row_data)

        # DataFrame from the populated data
        df_final = pd.DataFrame(column_values_data, columns=['Item ID', 'Item Name'
        dfs[board_name] = df_final  # Store the DataFrame in the dictionary using t
        # Display the final DataFrame
        #print(df_final.to_string(index=False))
    else:
        print(f"Failed to fetch data for board ID {board_id}: HTTP Status Code {res
```

In [4]:

```python
dfs['Deals']
```

	Item ID	Item Name	Annual Forecast Value	Annual Value	Client Address	Client City	Client Date of Birth	Client State	Client Zipcode	Close Date	...	Name	Patient
0	2431679	[redacted]										NaN	88
1	2497616											NaN	88
2	2497621											NaN	88

In [5]:

```python
# To CSV
dfs['Deals'].to_csv('monday_deals.csv', index=False)
```

Human Resource Tracking

HR KPIs Identified

Applicant ID

Applicant Name

Marketing Channel (Job Source)

Interviewed?

Offer Sent?

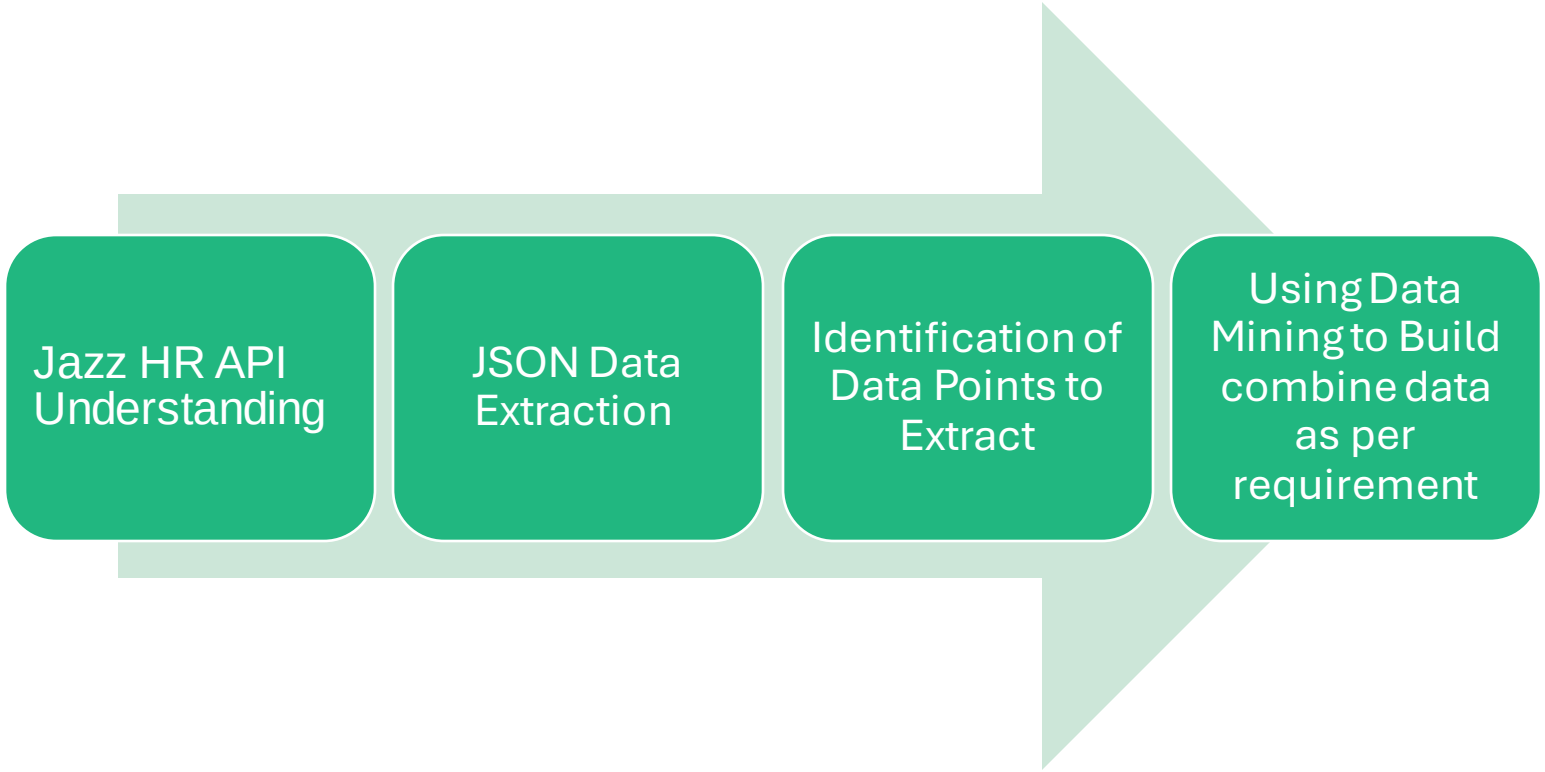
Accepted/Rejected Offer?

How long did they stay with us?

How many hours did they work?

Human Resource Tracking

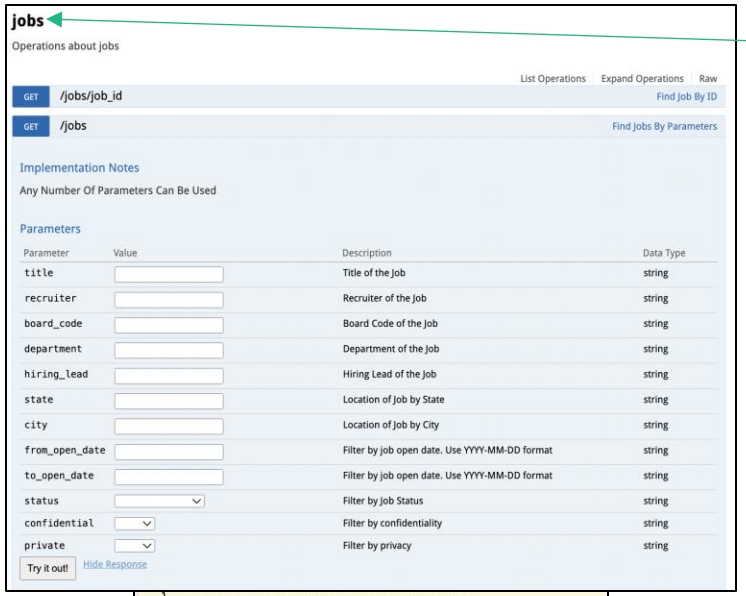
Steps Followed



Human Resource Tracking - Steps Followed

Jazz HR API Understanding

(1) Job Information



jobs

Operations about jobs

List Operations | Expand Operations | Raw

GET /jobs/job_id — Find Job By ID

GET /jobs — Find Jobs By Parameters

Implementation Notes

Any Number Of Parameters Can Be Used

Parameters

Parameter	Value	Description	Data Type
title		Title of the Job	string
recruiter		Recruiter of the Job	string
board_code		Board Code of the Job	string
department		Department of the Job	string
hiring_lead		Hiring Lead of the Job	string
state		Location of Job by State	string
city		Location of Job by City	string
from_open_date		Filter by job open date. Use YYYY-MM-DD format	string
to_open_date		Filter by job open date. Use YYYY-MM-DD format	string
status		Filter by Job Status	string
confidential		Filter by confidentiality	string
private		Filter by privacy	string

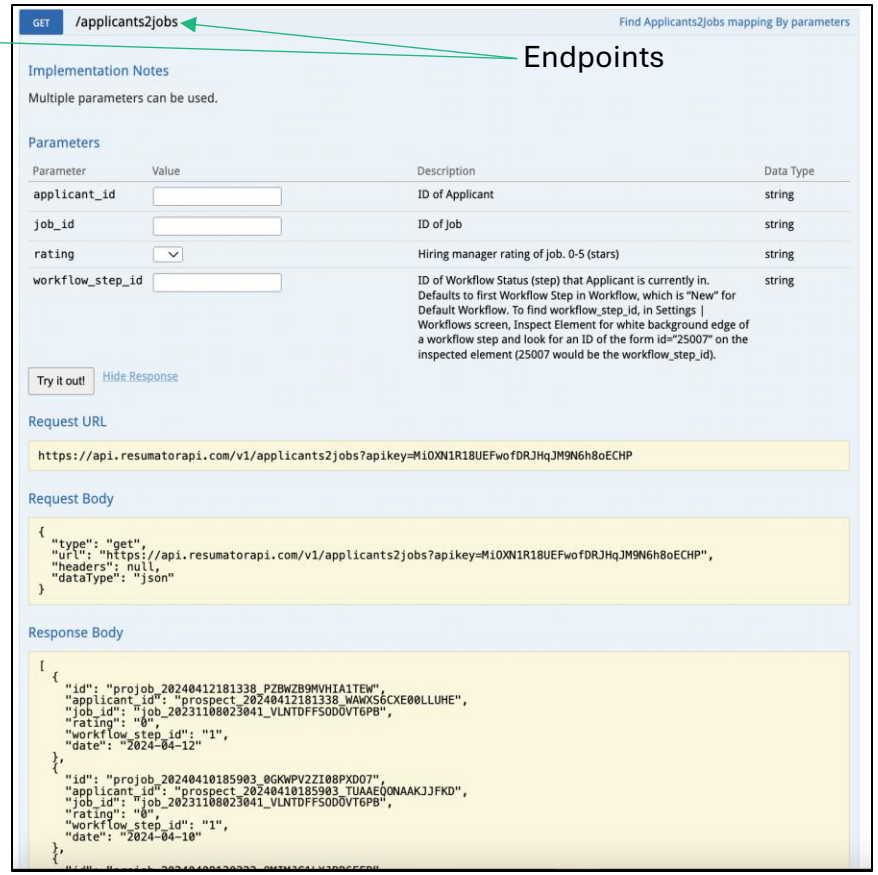
Try it out! Hide Response


```
  "id": "job_20231027120818_IWQSJVJR8COFQHJL",
  "title": "Sample Job",
  "country_id": "United States",
  "city": "Pittsburgh",
  "state": "PA",
  "zip": "15212",
  "department": "",
  "description": "
You can enter a detailed, formatted description of your position.

We have a fully featured WYSIWYG editor which enables you to craft

",
  "minimum_salary": "0",
  "maximum_salary": "0",
  "notes": "",
  "original_open_date": "2023-10-27",
  "type": "Full Time",
  "status": "On Hold",
  "send_to_job_boards": "No answer",
  "hiring_lead": "usr_20231027120817_BYLQIXJJFVWXRMCX",
  "board_code": "ZpTyI9FiNl",
  "internal_code": "",
  "questionnaire": "0"
 }
]
```


(2) Applicant's job application Information



GET /applicants2jobs — Find Applicants2Jobs mapping By parameters

Implementation Notes

Multiple parameters can be used.

Parameters

Parameter	Value	Description	Data Type
applicant_id		ID of Applicant	string
job_id		ID of Job	string
rating		Hiring manager rating of job. 0-5 (stars)	string
workflow_step_id		ID of Workflow Status (step) that Applicant is currently in. Defaults to first Workflow Step in Workflow, which is "New" for Default Workflow. To find workflow_step_id, in Settings | Workflows screen, Inspect Element for white background edge of a workflow step and look for an ID of the form id="25007" on the inspected element (25007 would be the workflow_step_id).	string

Try it out! Hide Response

Request URL

```
https://api.resumatorapi.com/v1/applicants2jobs?apikey=MiOXN1R18UEFwofDRJHqJM9N6h8oECHP
```

Request Body

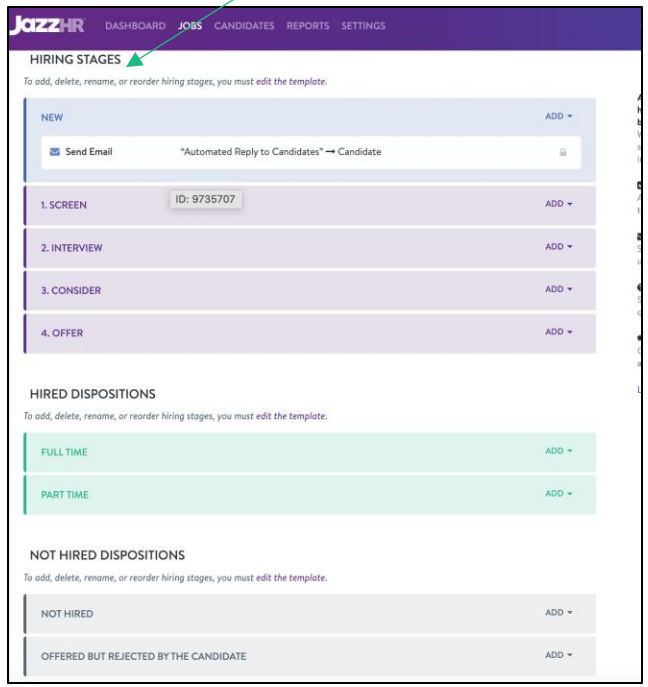

```
{
  "type": "get",
  "url": "https://api.resumatorapi.com/v1/applicants2jobs?apikey=MiOXN1R18UEFwofDRJHqJM9N6h8oECHP",
  "headers": null,
  "dataType": "json"
}
```


Response Body


```
[
 {
  "id": "projob_20240412181338_PZBWZB9MVHIA1TEW",
  "applicant_id": "prospect_20240412181338_WAWXS6CXE00LLUHE",
  "job_id": "job_20231108023041_VLNTDFFSODOVT6PB",
  "rating": "0",
  "workflow_step_id": "1",
  "date": "2024-04-12"
 },
 {
  "id": "projob_20240410185903_0GKWPV2ZI08PXDO7",
  "applicant_id": "prospect_20240410185903_TUAAEQONAAKJJFKD",
  "job_id": "job_20231108023041_VLNTDFFSODOVT6PB",
  "rating": "0",
  "workflow_step_id": "1",
  "date": "2024-04-10"
 },
 {
```


Endpoints

Applicant Job Application Workflow



JazzHR — DASHBOARD | JOBS | CANDIDATES | REPORTS | SETTINGS

HIRING STAGES

To add, delete, rename, or reorder hiring stages, you must edit the template.

NEW — ADD

Send Email — "Automated Reply to Candidates" → Candidate

1. SCREEN — ID: 9735707 — ADD

2. INTERVIEW — ADD

3. CONSIDER — ADD

4. OFFER — ADD

HIRED DISPOSITIONS

To add, delete, rename, or reorder hiring stages, you must edit the template.

FULL TIME — ADD

PART TIME — ADD

NOT HIRED DISPOSITIONS

To add, delete, rename, or reorder hiring stages, you must edit the template.

NOT HIRED — ADD

OFFERED BUT REJECTED BY THE CANDIDATE — ADD

Human Resource Tracking - Steps Followed

JSON Data Extraction

Request URL -

https://api.resumatorapi.com/v1/jobs?apikey=[redacted]

```
# Base URL for the API
base_url = "https://api.resumatorapi.com/v1"
api_key = "[redacted]"

# Function to fetch data from a specific endpoint
def fetch_data_from_endpoint(endpoint):
    # request URL
    url = f"{base_url}/{endpoint}?apikey={api_key}"

    # GET request
    response = requests.get(url)

    if response.status_code == 200:
        # Converting to pandas DataFrame
        data = response.json()
        df = pd.DataFrame(data)
        return df
    else:
        print(f"Failed to fetch data: {response.status_code}, Response: {response.text}")
        return None

# jobs usage
endpoint = 'jobs'
df = fetch_data_from_endpoint(endpoint)

if df is not None:
    df.to_csv('fetched_jobs.csv', index=False)
else:
    print("No data fetched.")

df
```

	id	title	country_id	city	state	zip	department	description	minimum_salary	maximum_salary
0	job_20231108023041_VLNTDFFSODOVT6PB	In Home Caregiver - Weekend Shifts	United States	Sharon	MA	02067		<p style="line-height:1.38;">Angel Care Associ...	0	0
1	job_20231027120818_IWQSJVJR8COFQHJL	Sample Job	United States	Pittsburgh	PA	15212		<p>You can enter a detailed, formatted descrip...	0	0

Other Endpoints in JazzHR API -

- activities
- applicants
- applicants2jobs
- hires
- categories
- contacts
- files
- jobs
- Tasks

Etc.

Human Resource Tracking - Steps Followed

Data Mining

Fetch the applicants with workflow step as hired and interviewed

For all the hired and interviewed candidates, set value as 'Yes' in the new columns defined

```
# applicants2jobs endpoint
endpoint = 'applicants2jobs'
df = fetch_data_from_endpoint(endpoint)

if df is not None:
    #print(df.head())
    applicants2jobs_df = df
    print("Columns in applicants:", applicants2jobs_df.columns)
    # Filter for hired and interviewed candidates
    hired_candidates = applicants2jobs_df[applicants2jobs_df['workflow_step_id'] == 'workflow_step_hired']
    interview_candidates = applicants2jobs_df[applicants2jobs_df['workflow_step_id'] == 'workflow_step_interview']

    # Identify hired/interviewed applicants by the workflow step ID
    hired_applicant_ids = hired_candidates['applicant_id'].unique()
    interview_applicant_ids = interview_candidates['applicant_id'].unique()

    # new column created in applicants_df that indicates whether the applicant is hired or interviewed
    applicants_df['hired_candidates'] = applicants_df['id'].apply(lambda x: 'Yes'
                                                                  if x in hired_applicant_ids
                                                                  else 'No')
    applicants_df['interview_candidates'] = applicants_df['id'].apply(lambda x: 'Yes'
                                                                      if x in interview_applicant_ids
                                                                      else 'No')
    applicants_df.to_csv('fetched_applicants2jobs.csv', index=False)

else:
    print("No data fetched.")

applicants_df
```

```
: Index(['id', 'applicant_id', 'job_id', 'rating', 'workflow_step_id', 'date'], dtyp
```

id	first_name	last_name	prospect_phone	apply_date	job_id	job_title	hired_candidates	interview_candidates
UMUDHO1	Abigail	Ogunleye	2348067335797	2024-03-26	job_20231108023041_VLNTDFFSODOVT6PB	In Home Caregiver - Weekend Shifts	No	No
MBZGQIBT	Odney	Dolcine	18572696161	2024-03-19	job_20231108023041_VLNTDFFSODOVT6PB	In Home Caregiver - Weekend Shifts	No	No
10OREEBL	Jennifer	Bezanson	18185165362	2024-03-19	job_20231108023041_VLNTDFFSODOVT6PB	In Home Caregiver -	No	No

Dashboard

Outcome

- The presented dashboard addresses the business questions regarding Recruitment and Retention outlined in the HCP Benchmarking Report.
- This signifies the first stage for Ankota in delivering efficient solutions to their clients for report completion, aiming to develop dashboards that tackle all inquiries.

RECRUITMENT & RETENTION QUESTIONS

Home Care **Top Caregiver Recruitment Sources and Methods** - Input your top two most effective caregiver recruitment sources and methods used in 2023.

Home Care **a) Caregiver Applications** - Did your business track the total number of caregivers who applied for employment in 2023?

☐ Yes ☐ No ☐ Do not know

b) Number of Caregiver Applications - How many caregiver employment applications did your business receive in 2023?

Number of Caregiver Applications in 2023: _____

Home Care **a) Caregiver Interviews** - Did your business track the number of caregiver interviews conducted for employment in 2023?

☐ Yes ☐ No ☐ Do not know

b) Number of Caregiver Interviews - How many caregiver interviews for employment did your business conduct in 2023?

Number of Caregiver Interviews in 2023: _____

Home Care **Caregivers Hired** - How many new caregivers were hired in 2023? *Whole numbers only.*

Caregivers Hired in 2023: _____

Home Care **Caregivers Employed** - How many caregivers were actively employed by your business for each of the following years? *(Used in caregiver turnover calculation.) Whole numbers only.*

Number of caregivers employed as of 12/31/2022: _____

Number of caregivers employed as of 12/31/2023: _____

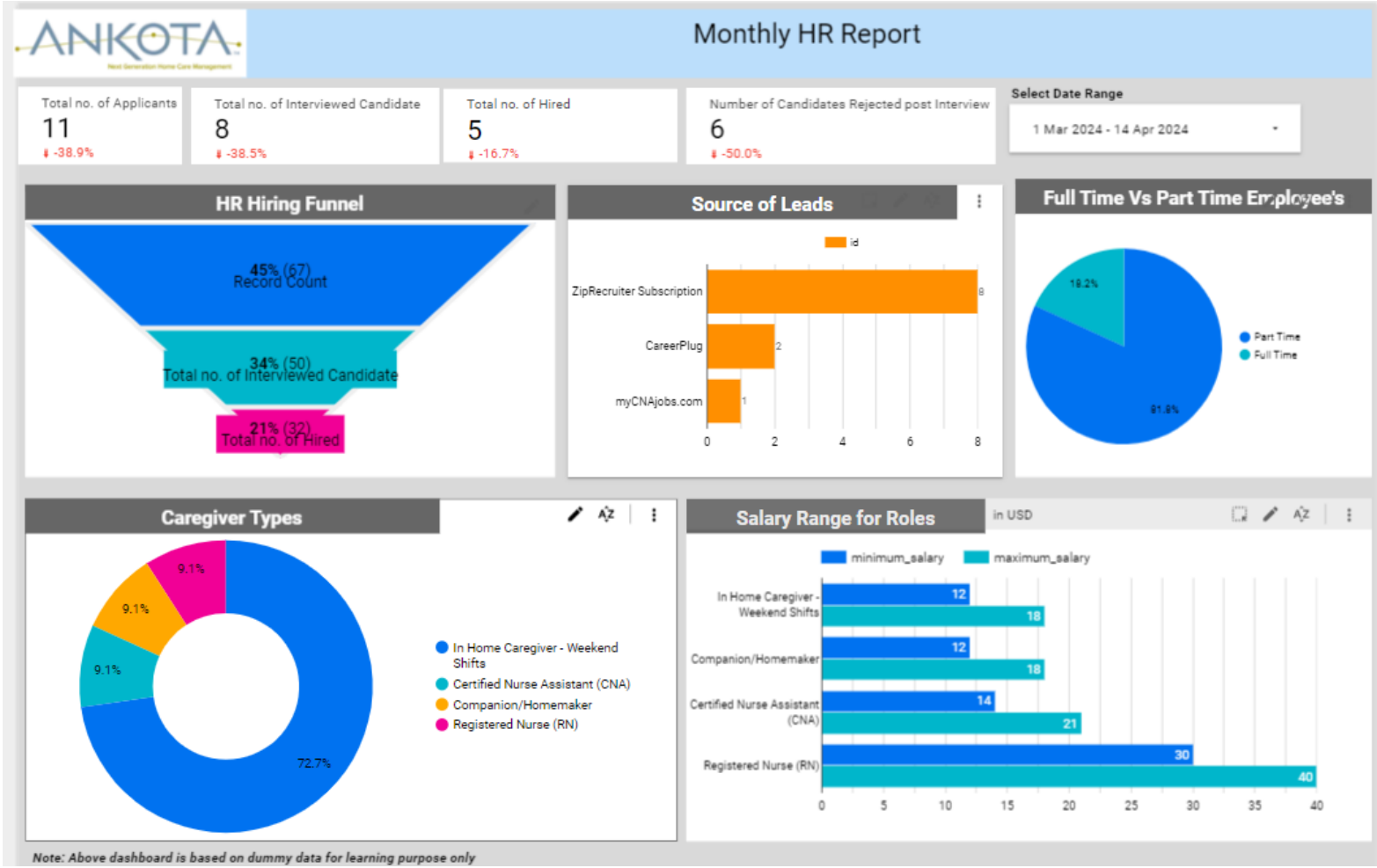
Home Care Home Health Hospice **Care Professional Hourly Wages** - What was your average starting HOURLY wage as of January 2023 for the following types of professional caregivers?

Leave any fields that do not apply blank.

2023	
_____	Companion/Homemaker
_____	Personal Care Attendant*
_____	Certified Nurse Assistant (CNA)
_____	Registered Nurse (RN)

HCP Benchmarking Report
Home-Based Care
Home Care, Home Health, and Hospice

HR Dashboard



Link to dashboard: https://lookerstudio.google.com/reporting/4aef7b20-2b9c-4633-87d3-42e97be24b5a/page/JHqwD

Learnings & Challenges

Translating business need into Business intelligence solution

Developing data to insight pipeline which includes data extraction, ETL and Data Visualization to deliver user friendly output.

Exploring new open-source tools like Looker, python for extraction and ETL

Identify operational KPIs for Healthcare agency (HR domain)


THANK YOU
ANKOTA PROJECT