<Data Engineering>

(CSL234)

Project Report

Faculty name: Aarti kukreja

Student name: Sakshi

Roll No.: 21csu419

Semester: 5th

Group: DS-B

Department of Computer Science and Engineering

The NorthCap University, Gurugram- 122001, India

Session 2023-24

PROJECT 1: POSTRESQL

PROJECT 2: POWER BI

PROJECT 1: POSTGRESQL

Table of Contents

PROJECT DESCRIPTION

The project, titled "Music Data Analytics," is a comprehensive exploration of a database designed for a music streaming service. The system incorporates data related to artists, albums, singles, users, playlists, and tracks. The primary goal of the project is to facilitate data engineering tasks and enable the extraction of valuable insights through SQL queries on a PostgreSQL database.

PROBLEM STATEMENT

The music industry generates vast amounts of data, and efficiently managing and extracting insights from this data is crucial for a music streaming service. The project aims to address the following challenges:

1. **Data Organization:** Efficiently organize data related to artists, albums, singles, users, playlists, and tracks to support analytical queries.
2. **Query Performance:** Optimize the database structure and queries to ensure quick and efficient retrieval of information.
3. **User Engagement:** Analyse user-related data to understand user behaviours, preferences, and engagement with playlists and tracks.
4. **Content Diversity:** Explore the diversity of music content by analysing the number of albums, singles, and tracks per artist.

ANALYSIS

The data engineering project involves the creation of tables for artists, albums, singles, users, playlists, and tracks. The dataset is then populated with realistic data using SQL INSERT statements. Various SQL queries are employed to perform data analysis, including joins between tables and aggregation functions.

HARDWARE REQUIREMENTS

The project has modest hardware requirements:

1. **Processor:** Dual-core processor or higher
2. **RAM:** 4 GB or higher
3. **Storage:** 20 GB of free disk space

SOFTWARE REQUIREMENTS

1. **Database Management System (DBMS):** PostgreSQL
2. **Querying Tool:** PostgreSQL command line or any SQL client supporting PostgreSQL
3. **Programming Language:** SQL for database queries

DESIGN

The database is designed with normalization principles, ensuring data integrity, and minimizing redundancy. Tables are appropriately linked using foreign key relationships to establish associations between entities. This design facilitates efficient querying and retrieval of information.

```sql
CREATE TABLE artists (
    artist_id SERIAL PRIMARY KEY,
    artist_name VARCHAR(100)
);
```

	artist_id [PK] integer	artist_name character varying (100)
1	1	Artist A
2	2	Artist B
3	3	Artist C
4	4	Artist D
5	5	Artist E

```sql
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    username VARCHAR(50),
    registration_date DATE,
    active_subscriber boolean,
    user_location VARCHAR(50)
);
```

	user_id [PK] integer	username character varying (50)	registration_date date	active_subscriber boolean	user_location character varying (50)
1	1	user1	2022-01-04	true	Denmark
2	2	user2	2022-01-29	true	Italy
3	3	user3	2022-01-29	false	Australia
4	4	user4	2022-01-06	false	United Kingdom
5	5	user5	2022-01-23	true	Australia
6	6	user6	2022-01-16	true	United Kingdom
7	7	user7	2022-01-02	true	Australia
8	8	user8	2022-01-13	false	Denmark

Total rows: 50 of 50 Query complete 00:00:00.104

```sql
CREATE TABLE albums (
    album_id SERIAL PRIMARY KEY,
    artist_id INT REFERENCES artists(artist_id),
    album_title VARCHAR(200),
    no_of_tracks INT,
    release_date DATE
);
```

	album_id [PK] integer	artist_id integer	album_title character varying (200)	no_of_tracks integer	release_date date
1	1	1	Album 1	8	2023-01-05
2	2	5	Album 2	6	2023-01-08
3	3	3	Album 3	6	2023-01-15
4	4	1	Album 4	9	2023-01-26
5	5	1	Album 5	10	2023-01-15
6	6	2	Album 6	9	2023-01-28
7	7	2	Album 7	6	2023-01-26
8	8	5	Album 8	9	2023-01-01

Total rows: 100 of 100 Query complete 00:00:00.164

```sql
CREATE TABLE singles (
    single_id SERIAL PRIMARY KEY,
    artist_id INT REFERENCES artists(artist_id)
    single_title VARCHAR(200),
    no_of_tracks INT,
    release_date DATE
);
```

	single_id [PK] integer	artist_id integer	single_title character varying (200)	no_of_tracks integer	release_date date
1	1	5	Single 1	3	2023-01-13
2	2	2	Single 2	3	2023-01-01
3	3	3	Single 3	3	2023-01-28
4	4	4	Single 4	3	2023-01-30
5	5	4	Single 5	3	2023-01-22
6	6	3	Single 6	3	2023-01-24
7	7	3	Single 7	2	2023-01-04
8	8	2	Single 8	2	2023-01-28

Total rows: 250 of 250 Query complete 00:00:00.083

```sql
CREATE TABLE album_tracks (
    atrack_id SERIAL PRIMARY KEY,
    album_id INT REFERENCES albums(album_id),
    atrack_title VARCHAR(200),
    duration INT,
    plays INT
);
```

	atrack_id [PK] integer	album_id integer	atrack_title character varying (200)	duration integer	plays integer
1	1	8	Track 1	138	138220
2	2	83	Track 2	141	303767
3	3	100	Track 3	356	187603
4	4	70	Track 4	199	45091
5	5	55	Track 5	135	231843

Total rows: 1000 of 1000 Query complete 00:00:00.085

```sql
CREATE TABLE single_tracks (
    strack_id SERIAL PRIMARY KEY,
    single_id INT REFERENCES singles(single_id),
    strack_title VARCHAR(200),
    duration INT,
    plays INT
);
```

	strack_id [PK] integer	single_id integer	strack_title character varying (200)	duration integer	plays integer
1	1	30	Track 1	355	651902
2	2	94	Track 2	185	454963
3	3	69	Track 3	376	183643
4	4	7	Track 4	130	326660
5	5	50	Track 5	239	265066

Total rows: 1000 of 2000 Query complete 00:00:00.093

```sql
CREATE TABLE playlists (
    playlist_id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(user_id),
    playlist_name VARCHAR(100),
    status  VARCHAR(10)
);
```

	playlist_id [PK] integer	user_id integer	playlist_name character varying (100)	status character varying (10)
1	1	22	Playlist 1	Private
2	2	47	Playlist 2	Public
3	3	40	Playlist 3	Private
4	4	38	Playlist 4	Public
5	5	2	Playlist 5	Public

Total rows: 100 of 100 Query complete 00:00:00.078

```sql
CREATE TABLE playlist_tracks (
    playlist_track_id SERIAL PRIMARY KEY,
    playlist_id INT REFERENCES playlists(playlist_id),
    atrack_id INT REFERENCES album_tracks(atrack_id),
    strack_id INT REFERENCES single_tracks(strack_id),
    added_at TIMESTAMP
);
```

	playlist_track_id [PK] integer	playlist_id integer	atrack_id integer	strack_id integer	added_at timestamp without time zone
1	1	63	100	653	2023-08-05 16:54:02.430408
2	2	6	666	[null]	2023-08-21 21:45:47.0334
3	3	44	954	510	2023-08-15 15:12:05.28984
4	4	68	673	524	2023-07-26 23:06:13.582385
5	5	30	[null]	464	2023-08-23 08:53:51.577803

Total rows: 1000 of 1000 Query complete 00:00:00.084

IMPLEMENTATION AND TESTING

The implementation involves the creation of tables, data insertion, and the execution of various SQL queries to derive meaningful insights. Testing includes verifying the accuracy of query results, assessing query performance, and ensuring the database can handle large datasets.

QUERIES results:

(the code will be included in the attachment)

1. **Find Number of Albums per Artist:**
    - This query counts the number of albums per artist, providing insights into the distribution of albums among different artists.
2. **Find the Artist with the Highest Number of Albums:**
    - Identifies the artist with the highest number of albums, indicating the most prolific artist in terms of album production.
3. **Find the Artist with the Highest Number of Singles:**
    - Determines the artist with the highest number of singles, showcasing the artist with a significant presence in the singles category.
4. **Find the Album with the Maximum Number of Tracks:**
    - Identifies the album with the maximum number of tracks, offering insights into the album with the most extensive track list.
5. **Find the Single with the Maximum Number of Tracks:**
    - Identifies the single with the maximum number of tracks, highlighting the single with the most diverse content.
6. **Find the Average Album Duration:**
    - Calculates the average duration of tracks in albums, providing an overview of the typical length of album tracks.

7. **Find the Average Single Duration:**
    - Calculates the average duration of tracks in singles, offering insights into the typical length of individual tracks.
8. **Find the Average Number of Track Plays:**
    - Computes the average number of plays for both album and single tracks, giving an overview of the average popularity of tracks in each category.
9. **Find the Album and Singles Count per Artist:**
    - This complex query combines counts of albums and singles per artist, providing a holistic view of an artist's contribution to both categories. The results are sorted by the total count, and the top 5 artists are retrieved.
10. **Users per Country:**

- Counts the number of users per country, offering insights into the distribution of users across different locations.

11. **Total Playlists Curated:**

- Counts the total number of playlists curated by users, providing insights into the engagement level of users in creating playlists.

12. **User Engagement:**

- Counts the number of new users registered over time, offering insights into user acquisition trends.

13. **Find Active vs. Inactive Users:**

- Classifies users as active or inactive and counts the number of users in each category, providing insights into user engagement.

14. **User Playlist Duration:**

- Calculates the total duration of playlists for each user, offering insights into the overall playlist duration created by users.

15. **Number of Public and Private Playlists:**

- Counts the number of public and private playlists, providing insights into the distribution of playlist types.

16. **Users with the Most Diverse Taste in Music:**

- Identifies users with the most unique tracks added to playlists, showcasing users with diverse musical preferences.

17. **Number of Blank Playlists:**

- Identifies playlists with no associated tracks, providing insights into the existence of empty or incomplete playlists.

18. **User Playlist Diversity:**

- Calculates the average distinct artist count in playlists for each user, offering insights into the diversity of musical preferences among users.

19. **Identify Users with Playlists Containing Tracks from the Same Artist:**

- Identifies users with playlists containing tracks from the same artist, indicating users with a preference for a specific artist in their playlists.

20. **Top 10 Played Tracks:**

- Retrieves the top 10 tracks based on the number of plays, providing insights into the most popular tracks.

21. **Preference Comparison between Albums and Singles:**

- Compares the total plays for albums and singles, offering insights into the relative popularity of the two categories.

22. **Top 5 Playlists with Most Tracks:**

- Retrieves the top 5 playlists with the highest number of tracks, providing insights into the most extensive playlists.

23. **Top 10 Popular Playlists:**

- Retrieves the top 10 playlists based on the number of tracks, offering insights into the most popular playlists.

24. **Top 5 Popular User Locations:**

- Counts the number of users in each location and retrieves the top 5 locations with the highest user count.

25. **Popular Playlists among Active Users:**

- Retrieves the top 5 playlists among active subscribers based on the number of tracks, providing insights into the preferences of active users.

26. **Location-Wise Active Users:**

- Counts the number of active subscribers in each location, offering insights into the distribution of active users across different regions.

27. **Location-Wise Most Popular Playlists:**

- Identifies the most popular playlists in each location based on the total track count, providing insights into regional playlist preferences.

28. **Location-Wise Most Popular Artists:**

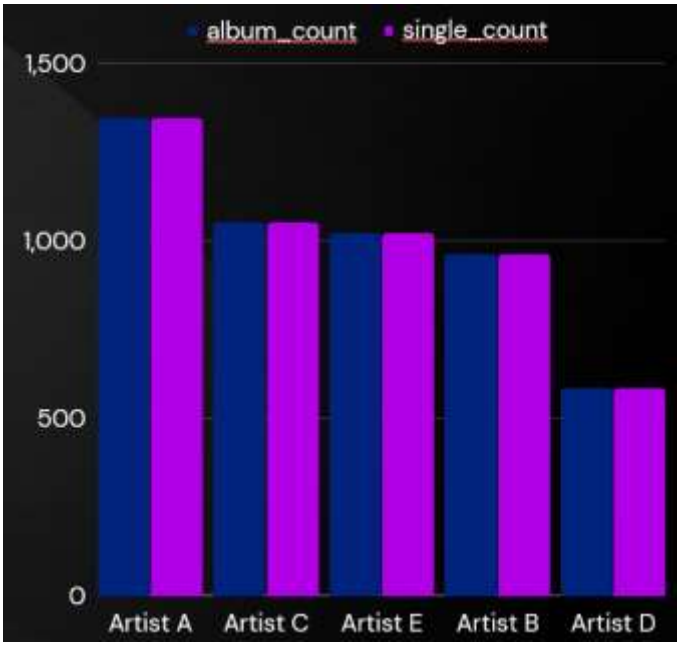
- Identifies the most popular artists in each location based on the total plays, offering insights into regional artist preferences.

OUTPUT

ARTIST'S ANALYSIS

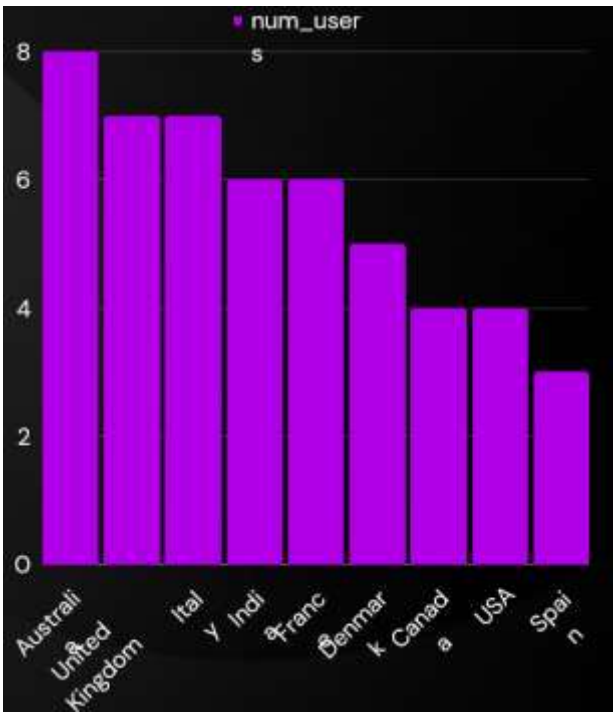
1. Location-wise popular artist

2. Count of albums/singles

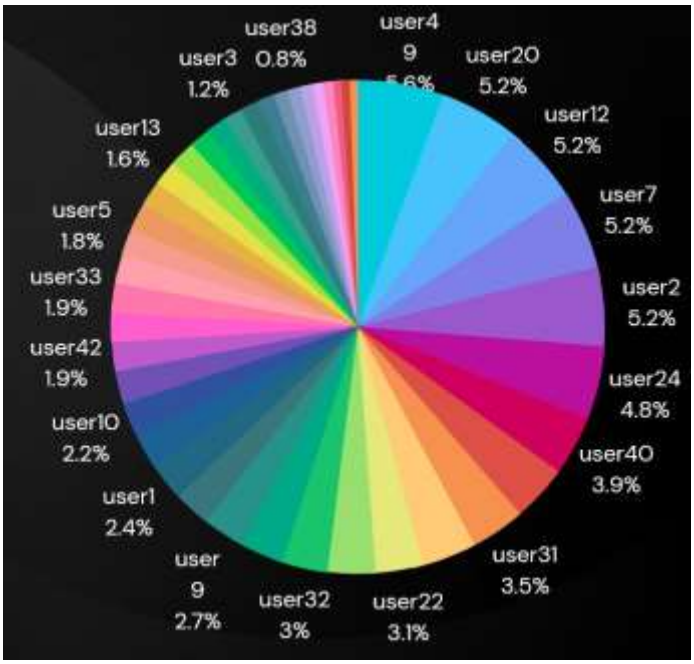


USER ANALYSIS

1. Country wise number of users.

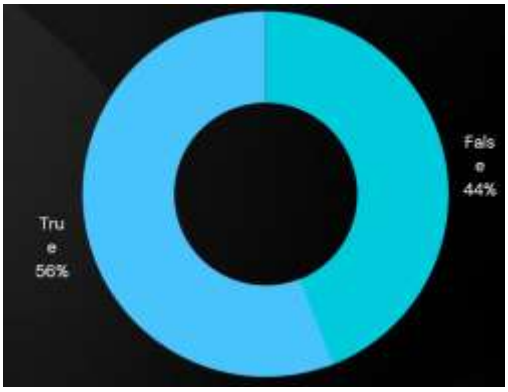


2. Users with diverse taste in music.

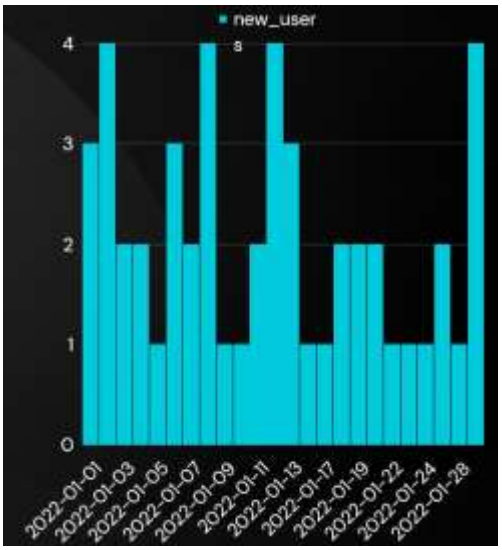


PERFORMANCE ANALYSIS

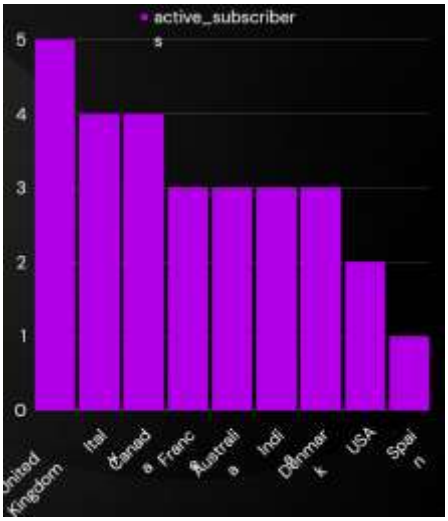
1. Active and Inactive users.



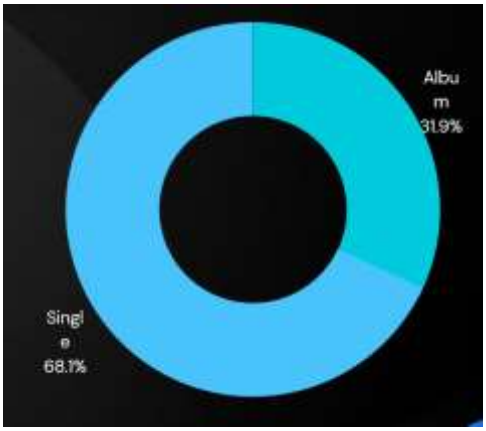
2. User engagement.



3. Location-wise active users.

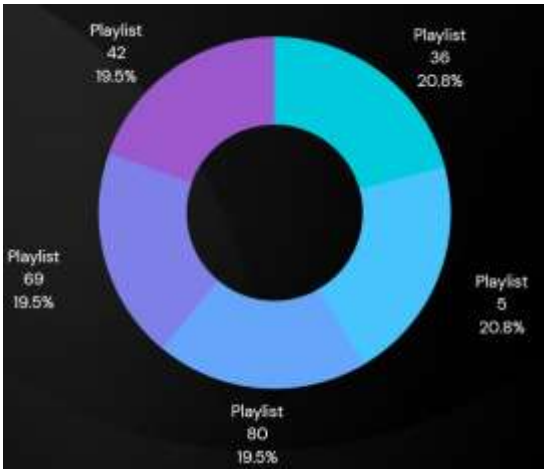


4. User preference between albums and singles.

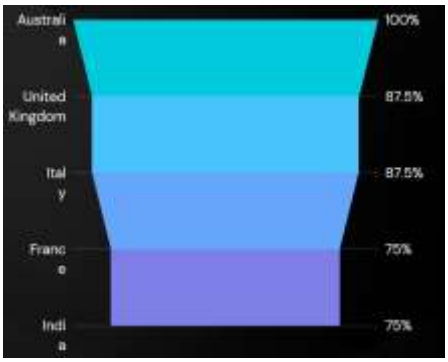


5. Public and Private playlists.

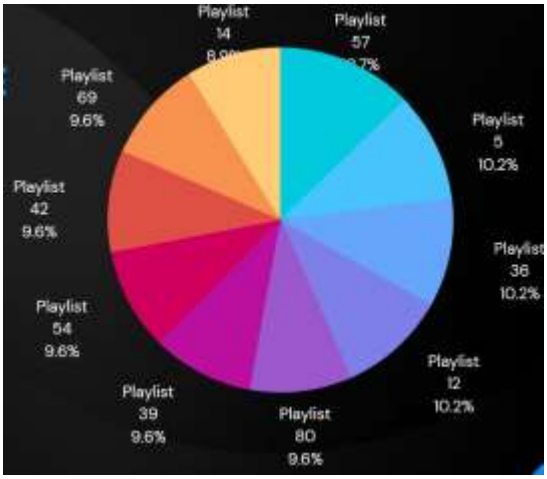
6. Popular playlists among active users.



7. Top 5 locations.



8. Top 10 playlists.



CONCLUSION AND FUTURE SCOPE

In conclusion, this web analytics project focused on the in-depth analysis of a music streaming website's data to derive valuable insights regarding user behavior, artist preferences, and playlist engagement. Through careful SQL queries and data exploration, we unveiled notable trends such as top-performing artists and tracks, user engagement patterns, and playlist diversity. The findings shed light on the impact of geographical locations on music preferences and allowed us to pinpoint opportunities for enhancing user experience and content curation. By understanding these analytics, the project equips stakeholders with actionable insights to optimize the website's performance, tailor marketing strategies, and foster a more engaging music streaming platform for users worldwide

PROJECT 2: POWER BI

Table of Contents

PROJECT DESCRIPTION

The project involves the exploration and analysis of a real-world job posting dataset using Power BI. The dataset is sourced from Data Search, a fictional recruitment company, and is aimed at extracting valuable insights to aid decision-making processes. The goal is to leverage Power BI's capabilities to create an interactive and informative business dashboard.

PROBLEM STATEMENT

The recruitment industry generates vast amounts of data daily, including job postings, candidate profiles, and market trends. Data Search aims to harness this data to gain a competitive edge, enhance decision-making, and streamline their operations. The specific challenges addressed include identifying trends in job demand, understanding candidate preferences, and optimizing recruitment strategies.

ANALYSIS

The analysis phase involves exploring the dataset to uncover patterns, trends, and correlations. Key areas of focus include:

1. **Job Market Trends:** Analysing the distribution of job postings across industries, locations, and experience levels.
2. **Candidate Preferences:** Understanding the skills and qualifications most sought after by candidates.
3. **Recruitment Efficiency:** Evaluating the time taken to fill different types of positions and identifying potential bottlenecks.
4. **Diversity and Inclusion:** Investigating diversity metrics in job postings to promote inclusivity.

HARDWARE REQUIREMENTS

- Standard desktop or laptop with a minimum of 8GB RAM.
- Adequate storage space for dataset and Power BI files.
- Internet connectivity for data sourcing (if applicable).

SOFTWARE REQUIREMENTS

- Microsoft Power BI Desktop.
- Dataset in a compatible format (CSV, Excel, etc.).
- Microsoft Office for additional documentation.

DESIGN

1. **Data Import:** Load the dataset into Power BI Desktop for analysis.
2. **Data Cleaning:** Address missing values, outliers, and any inconsistencies in the dataset.
3. **Data Modelling:** Create relationships between different tables in the dataset to enable effective analysis.
4. **DAX Formulas:** Utilize Data Analysis Expressions (DAX) to create calculated columns and measures for advanced analytics.
5. **Visualization:** Design visually appealing and insightful charts, graphs, and tables to represent key findings.

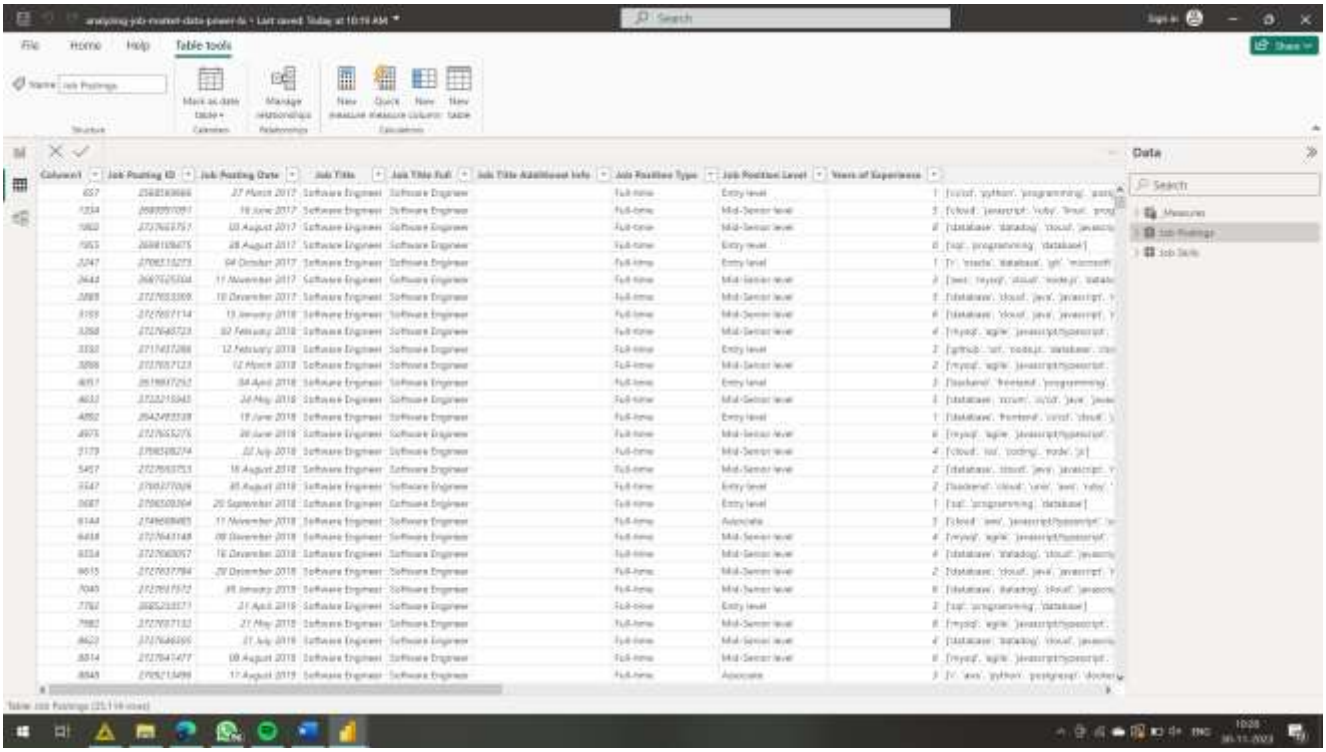
Job Details & Requirements	
Job Title	The simplified title for the job position (e.g., "Data Engineer")
Job Title Full	The full title for the job position (e.g., "Senior Azure Data Engineer")
Job Title Additional Info	Any additional information for a job title (e.g., "Remote", "[blank]")
Job Position Type	The time/job requirements: "Full-time", "Part-time", "Internship", "Contract"
Job Position Level	Indicates seniority of a job position: "Entry level", "Executive", etc
Years of Experience	The number of years of experience
Job Skills	List of skill requirements
Minimum Pay	The lowest salary/pay offered
Maximum Pay	The highest salary/pay offered
Pay Rate	The rate of pay for the job: hourly ("hr"), salary ("yr")

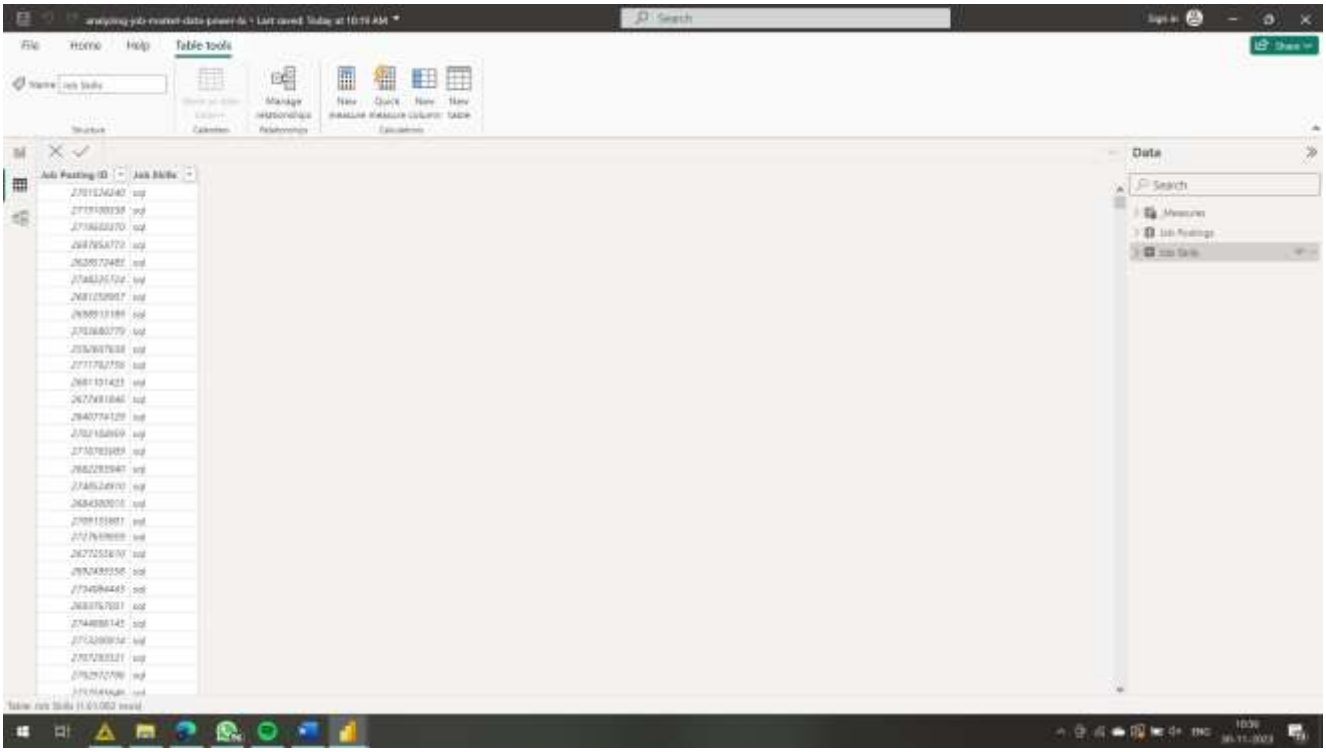
Job Posting Identification	
Job Posting ID	The unique identification number for each job posting
Job Posting Date	The date of the job posting
Number of Applicants	The number of those that applied for the job in the first 24 hours

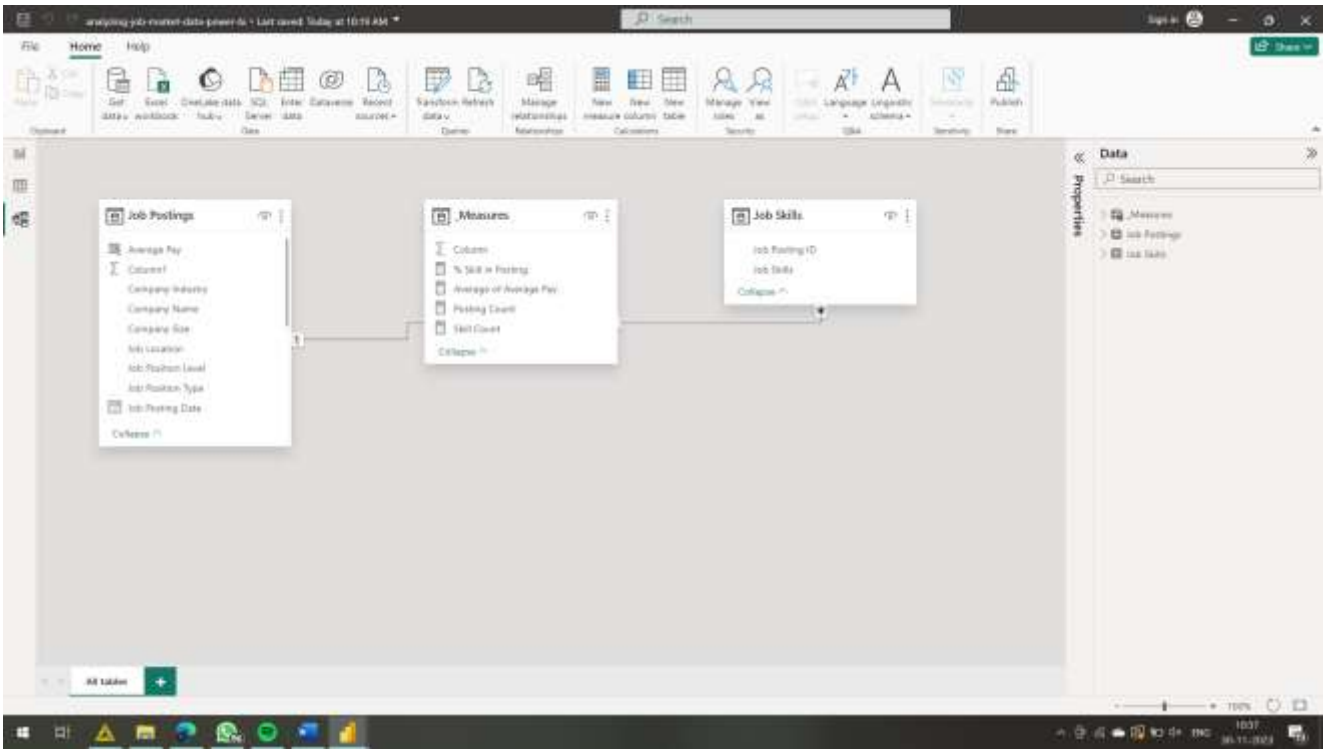
Company Details	
Company Name	The name of the company sponsoring the job
Company Industry	The industry that the company is involved
Company Size	The size of the company by the number of employees
Job Location	The geographic location of the company and job

IMPLEMENTATION AND TESTING

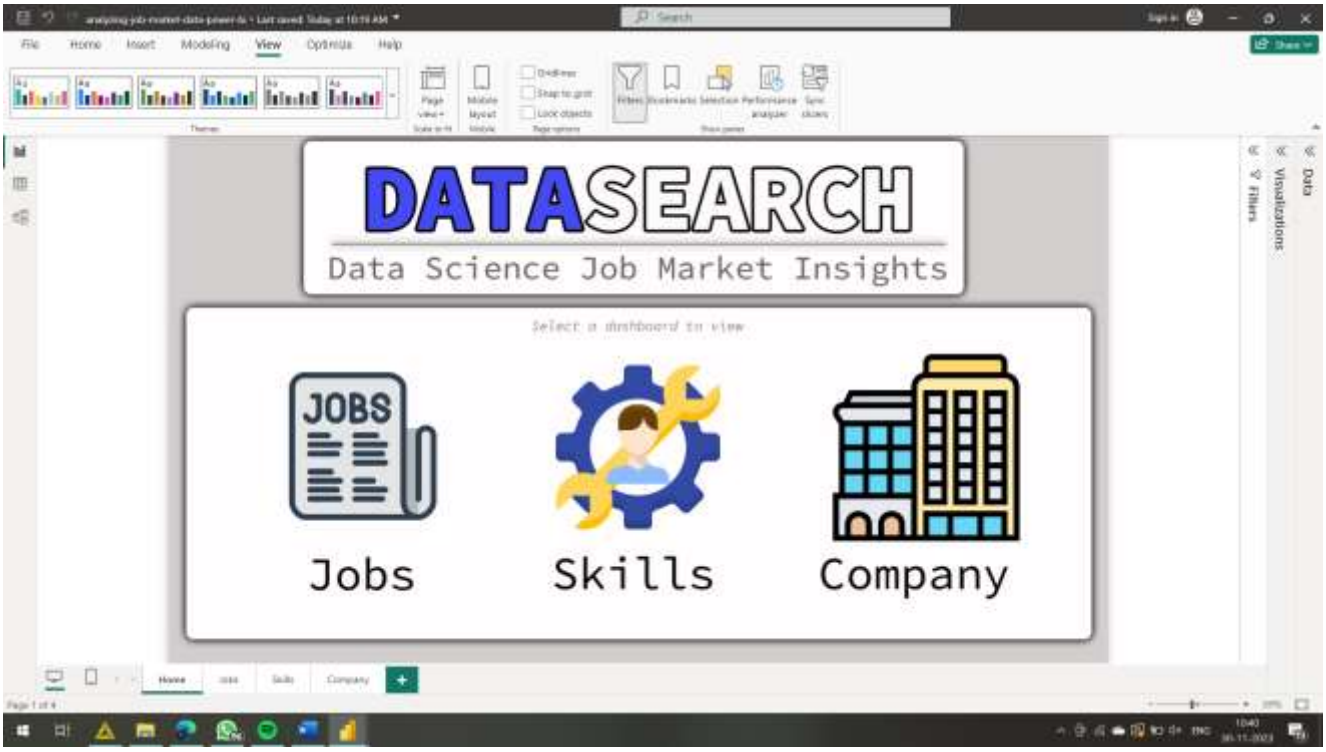
1. **Data Import and Cleaning:** Load the dataset into Power BI Desktop, addressing any issues with data quality.
2. **Data Modelling:** Establish relationships and create calculated fields using DAX.
3. **Visualization:** Design and implement the visualizations according to the predefined analysis areas.
4. **Dashboard Creation:** Aggregate visualizations into an interactive dashboard for a comprehensive view.
5. **Testing:** Validate the accuracy of visualizations and ensure the dashboard meets the defined objectives.

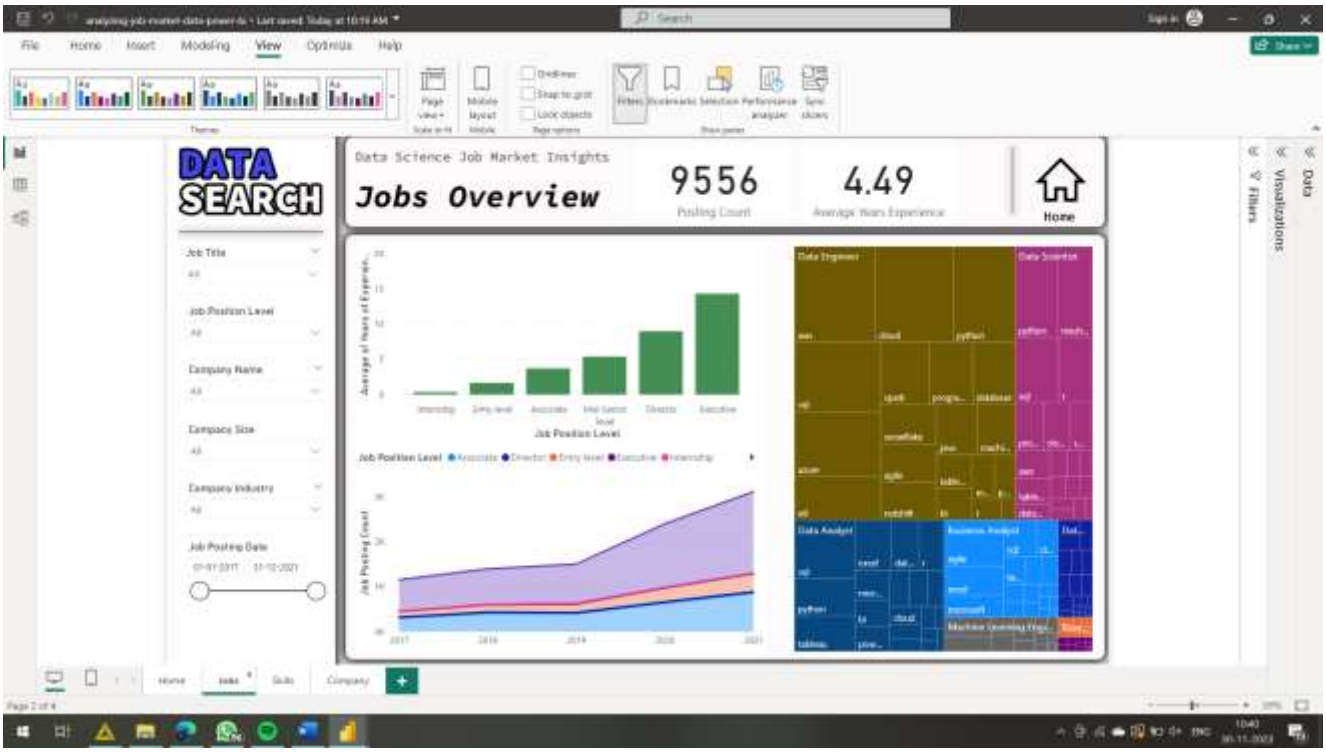


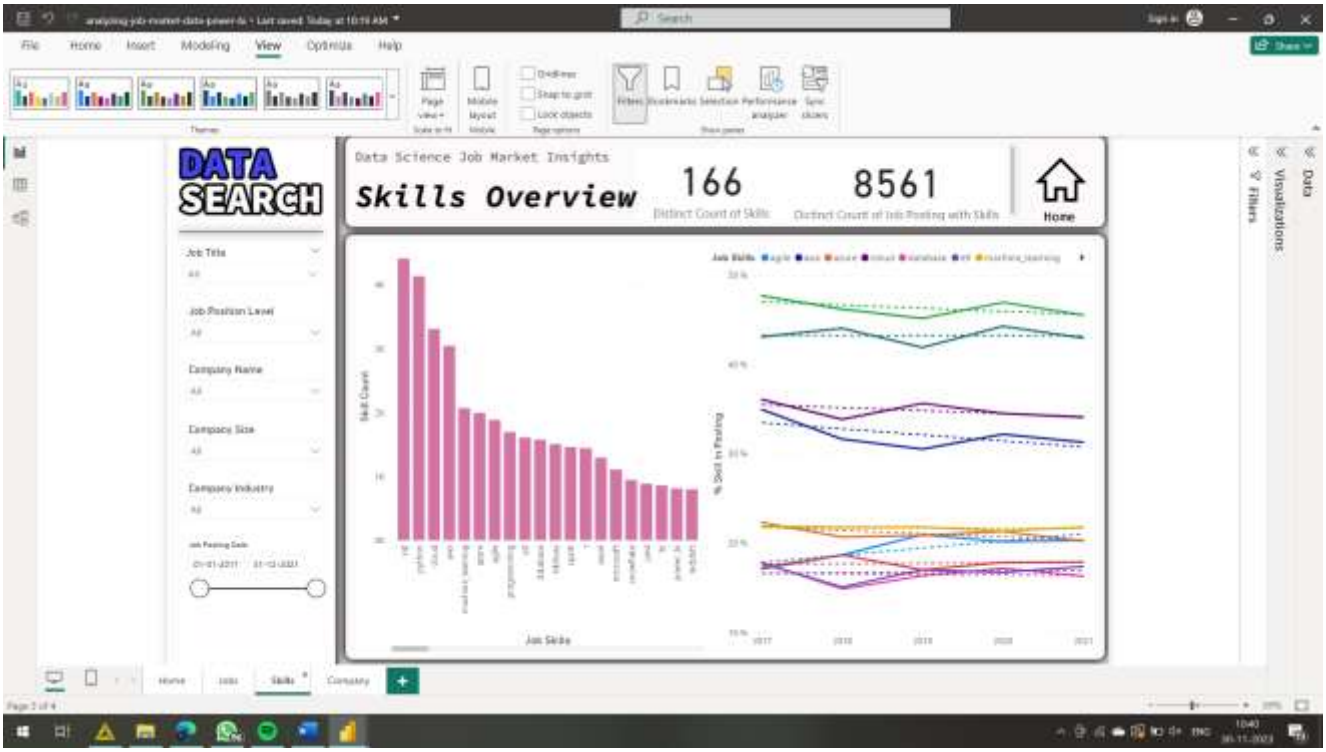

File
Home
Help
Table tools
Data
Search

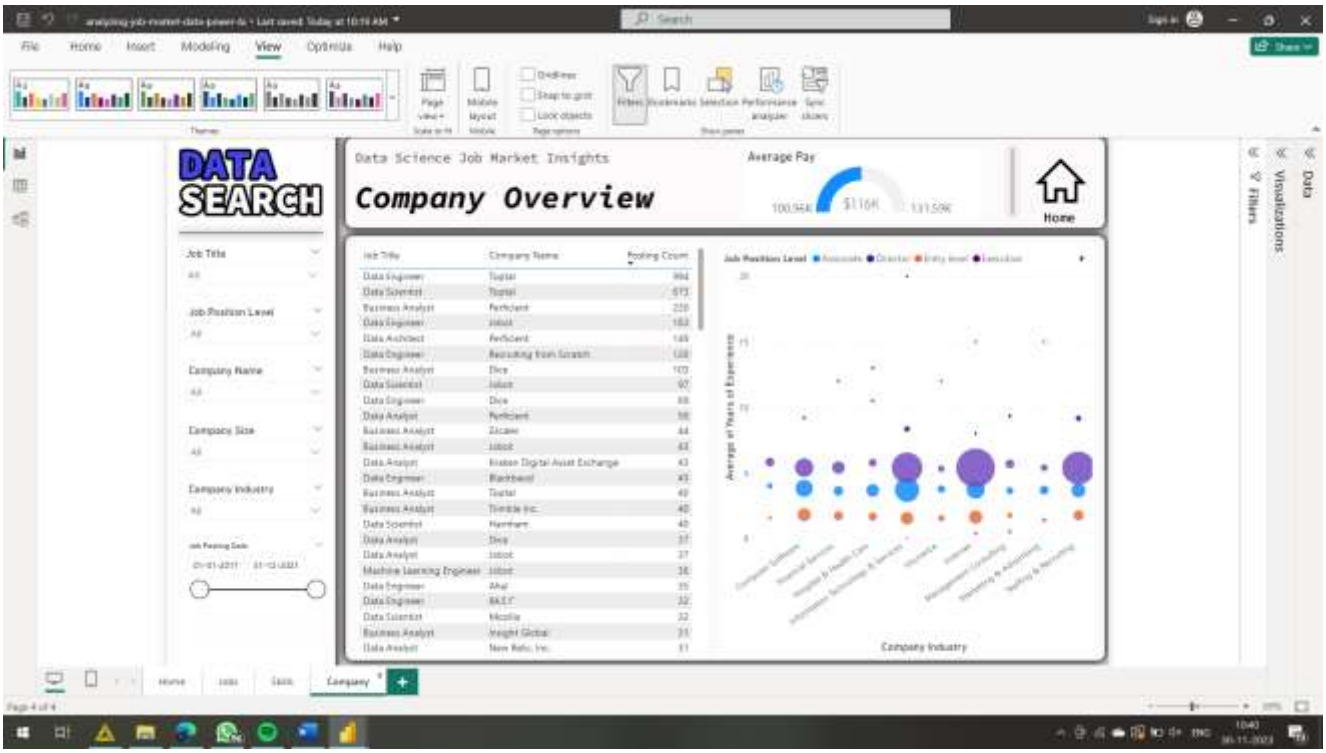

File
Home
Help
Job Postings
_Measures
Job Skills
Data
Search
Properties

OUTPUT


DATASEARCH
Data Science Job Market Insights
Select a dashboard to view
JOBS
Jobs
Skills
Company


DATA SEARCH
Data Science Job Market Insights
Jobs Overview
9556
Posting Count
4.49
Average Years Experience
Home
Job Title
Job Position Level
Company Name
Company Size
Company Industry
Job Posting Date


DATA SEARCH
Data Science Job Market Insights
Skills Overview
166
Distinct Count of Skills
8561
Distinct Count of Job Posting with Skills
Home
Job Title
Job Position Level
Company Name
Company Size
Company Industry
Skill Count
% Skill in Posting
Job Skills


DATA SEARCH
Data Science Job Market Insights
Company Overview
Average Pay
$116K
Home
Job Title
Company Name
Job Position Level
Company Size
Company Industry
Average of Years of Experience
Company Industry

CONCLUSION AND FUTURE SCOPE

In conclusion, this Power BI project enables Data Search to harness the power of data for strategic decision-making in the recruitment domain. The interactive dashboard serves as a valuable tool for quick and efficient analysis. The future scope of this project involves continuous updates to the dataset, integration with real-time data sources, and further enhancements based on evolving business requirements. Additionally, the project sets the foundation for leveraging advanced analytics and machine learning to enhance predictive modeling for recruitment outcomes.