**LEARNING IMPACT ON JOB PERFORMANCE AND RETENTION**

NEW WAVE FINAL PROJECT

JMD350 KISHORE

**LEARNING IMPACT ON JOB PERFORMANCE AND RETENTION**

NEW WAVE FINAL PROJECT

Contents

[1 Full Stack 2](#_Toc179820740)

[1.1 Overview 2](#_Toc179820741)

[1.2 Problem Statement 2](#_Toc179820742)

[1.3 Modules 2](#_Toc179820743)

[1.3.1 Login Module: 2](#_Toc179820744)

[1.3.2 Admin Module: 3](#_Toc179820745)

[1.3.3 Employee Module: 6](#_Toc179820746)

[1.4 Front-end Specifications 7](#_Toc179820747)

[1.5 Back-end Specifications 7](#_Toc179820748)

[1.6 Database Details 7](#_Toc179820749)

[1.7 API Documentation 8](#_Toc179820750)

[1.8 Architecture 13](#_Toc179820751)

[1.9 Future Roadmap 13](#_Toc179820752)

[2 Data Engineering 14](#_Toc179820753)

[2.1 Overview 14](#_Toc179820754)

[2.2 Problem Statement 14](#_Toc179820755)

[2.3 Architecture 14](#_Toc179820756)

[2.4 Reports 15](#_Toc179820757)

[2.5 Data Warehouse Structure 15](#_Toc179820758)

[2.6 Logging 16](#_Toc179820759)

[2.7 Conclusion 16](#_Toc179820760)

[3 Data Science 17](#_Toc179820761)

[3.1 Problem Statement 17](#_Toc179820762)

[3.2 EDA 17](#_Toc179820763)

[3.2.1 No. of. Unique Values 17](#_Toc179820764)

[3.2.2 Awards won on each Education category 17](#_Toc179820765)

[3.2.3 Promotion counts on each Education category 18](#_Toc179820766)

[3.2.4 Gender Distribution 18](#_Toc179820767)

[3.2.5 Length of Service by recruitment channel 19](#_Toc179820768)

[3.2.6 Age Distribution by education 19](#_Toc179820769)

[3.3 Data Preprocessing 19](#_Toc179820770)

[3.4 Feature Engineering 19](#_Toc179820771)

[3.5 Model Training 20](#_Toc179820772)

[3.6 Model Evaluation 20](#_Toc179820773)

[3.7 Conclusion 20](#_Toc179820774)

# Full Stack

## Overview

The primary objective of this project is to develop a comprehensive module that correlates learning outcomes with employee job performance and retention rates. By establishing a clear link between the training programs undertaken by employees and their subsequent performance on the job, organizations can make informed decisions about employee development and resource allocation.

Key components of the module include:

* Admin Dashboards: Visualizations that showcase the impact of learning activities on job performance and retention. These dashboards will include metrics such as top learners, course completion rates, and performance improvements.
* Performance Tracking: A dedicated page to capture and display the scores of training programs, enabling organizations to assess the effectiveness of their training initiatives.
* Retention Metrics: A systematic approach to tracking employee retention data alongside their learning history, helping identify trends and factors that influence employee retention.
* Database Schema: A structured database schema that stores vital information regarding employee performance metrics, retention data, and learning histories, ensuring efficient data management and retrieval.

## Problem Statement

Develop a module that correlates learning outcomes with employee job performance and retention.

**Design & Build:**

* Create admin dashboards that visualize the impact of learning activities on job performance and retention.
* Create a page to capture the scores of the training program.
* Create a page to display performance improvements post-training.
* Create a database schema to store employee performance metrics, retention data, and learning history.

## Modules

### Login Module:

Develop a module that correlates learning outcomes with employee job performance and retention.

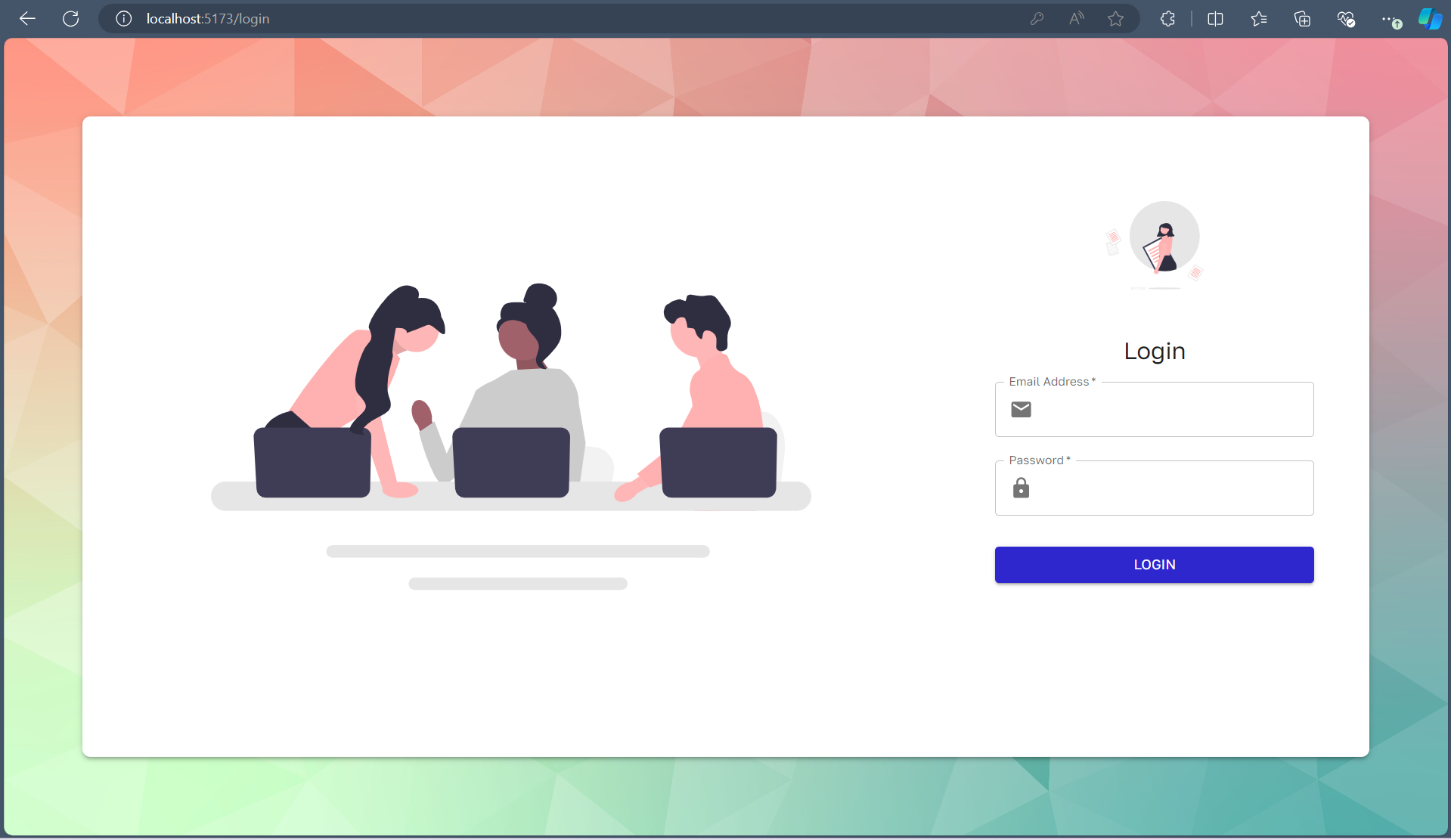
****

Figure Login Page

### Admin Module:

* **Dashboard:**
  + Top Learners: The employees who completed the most no. of courses is listed here.
  + Top Performers: The top 10 employees with high average score are listed here.
  + Course Data: Course and the designations that it is assigned to are listed here.
  + Percentage of Course Status: The course with its completion, failed and incomplete count are listed here.

A screenshot of a computer

Description automatically generated

Figure Admin Dashboard Page 1

A screenshot of a computer

Description automatically generated

Figure Admin Dashboard Page 2

A screenshot of a computer

Description automatically generated

Figure Admin Dashboard Page 3

* **Employee Report:**
  + This contains employee details with their course status count.

A screenshot of a computer screen

Description automatically generated

Figure Employees Page

* + **Detail Employee Report**:
    - Job Performance: This contains the feedback score of the employees’ project which are collected from the project manager’s feedback.
    - Course-wise Performance: Contains the course score that the employee has completed
    - Top Performance: Course with highest score.
    - Average Score: Average Score of all the completed courses.

A screenshot of a computer

Description automatically generated

Figure Employee Details Page 1

A screenshot of a computer

Description automatically generated

Figure Employee Details Page 2

* **Course Report:**
  + This contains Course details with employee status count

A screenshot of a computer

Description automatically generated

Figure Courses Page

* + **Detail Course Report**:
    - Top performance: The top 10 employees with high score are listed here.
    - Course Status Chart: Shows the course completed, failed and incomplete count.
    - Designation-wise Report: Shows the top and average score of each designation.

A screenshot of a computer

Description automatically generated

Figure Course Performance Report Page

### Employee Module:

* **Job Performance Report:**
  + This contains the feedback score of the employees’ project which is collected from the project manager’s feedback.

A screenshot of a computer

Description automatically generated

Figure Employee Job Performance Page

* **Course Report:**
  + Course-wise Performance: Contains the course score that the employee has completed
  + Top Performance: Course with highest score.
  + Average Score: Average Score of all the completed courses.

**A computer screen shot of a graph

Description automatically generated**

Figure Employee Course Details Page

## Front-end Specifications

**GitHub Repository:** [Kishore29-02/Final\_Project\_FS\_Client (github.com)](https://github.com/Kishore29-02/Final_Project_FS_Client)

**Framework:** [React JS](https://legacy.reactjs.org/)

**UI Component Library:** [Material UI](https://mui.com/material-ui/)

**Chart Library:** [Apex charts](https://apexcharts.com/)

**Build Tool:** [Vite](https://v2.vitejs.dev/guide/)

## Back-end Specifications

**GitHub Repository:** [Kishore29-02/Final\_Project\_FS\_Server (github.com)](https://github.com/Kishore29-02/Final_Project_FS_Server)

**Framework:** [Node JS](https://nodejs.org/en)

**Database ORM:** [Prisma](https://www.prisma.io/)

**Password Hashing:** [bcrypt](https://www.npmjs.com/package/bcryptjs)

**Web Server:** [Express](https://expressjs.com/)

## Database Details

**Database used**: [PostgreSQL](https://www.postgresql.org/)

**ER Diagram:**

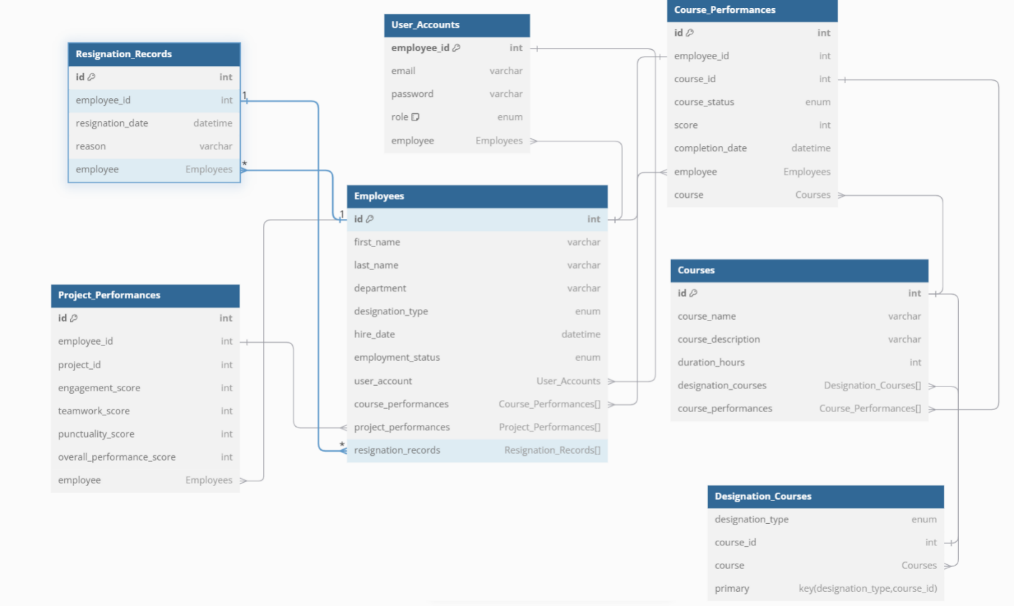
****

Figure ER Diagram

The diagram represents the relationships between the following entities:

* Employees
* User\_Accounts
* Courses
* Course\_Performances
* Project\_Performances
* Designation\_Records
* Resignation\_Records

**Entity 1: Employees**

* Attributes: id, first\_name, last\_name, department, designation\_type, hire\_date, employment\_status.
* Connected to both “User\_Accounts,” “Course\_Performances,” “Project\_Performances,” and “Resignation\_Records.”

**Entity 2: User\_Accounts**

* Attributes: employee\_id, email, password, role.
* Connected to the “Employees” entity.

**Entity 3: Courses**

* Attributes: id, course\_name, course\_description, duration\_hours.
* Connected to both “Designation\_Courses” and “Course\_Performances.”

**Entity 4: Designation\_Courses**

* Attributes: designation\_type, course\_id.
* Represents the relationship between specific designations and their corresponding courses.

**Entity 5: Course\_Performances**

* Attributes: id, employee\_id, course\_id, course\_status, score, completion\_date.
* Connected to the “Employees” and “Courses” entities.

**Entity 6: Project\_Performances**

* Attributes: id, employee\_id, project\_id, engagement\_score, teamwork\_score, punctuality\_score, overall\_performance\_score.
* Connected to the “Employees” entity.

**Entity 7: Resignation\_Records**

* Attributes: id, employee\_id, resignation\_date, reason.
* Connected to the “Employees” entity.

## API Documentation

**Endpoint:** /login

This endpoint authenticates a user by verifying their email and password and returns a JWT token for authorized access.

* Request-Type: Post
* Content-Type: application/json
* Request-Body :

{

"email": "string",

"password": "string"

}

* Response:
  + Success Response (**200 OK**):

{

"message": "Authentication successful",

"token": "jwt\_token\_string",

"id": "number",

"name": "string",

"role": "string"

}

* + Failure Response (**403 FORBIDDEN**):
    - Invalid email:

{

"message": "Email does not exist"

}

* + - Invalid Password:

{

"message": "Invalid password"

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /employee/?email=

Retrieves employee details along with their course performance and project count. Allows filtering by email.

* Method: Get
* Request-Parameters: email
* Response:
  + Success Response (**200 OK**):

[

{

"id": "number",

"firstName": "string",

"lastName": "string",

"dept": "string",

"designation": "string",

"email": "string",

"projectCount": "number",

"courseCompletedCount": "number",

"courseFailedCount": "number",

"courseIncompleteCount": "number"

}

]

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /employee/performance/?id=

Retrieves detailed performance data for an employee, including job performance, course assignments, and course scores.

* Method: Get
* Request-Parameters: id
* Response:
  + Success Response (**200 OK**):

{

"jobPerformance": "object",

"coursesAssigned": [

{

"courseName": "string",

"courseStatus": "string",

"score": "number",

"completionDate": "string"

}

],

"completedCourseScores": [

{

"name": "string",

"score": "number"

}

],

"topScoreAndCourse": {

"score": "number",

"courseName": "string"

},

"averageScore": "number"

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /employee/job-performance/?id=

Retrieves job performance data for a specific employee based on their ID.

* Method: Get
* Request-Parameters: id
* Request-Body:

{

"email": "string",

"password": "string"

}

* Response:
  + Success Response (**200 OK**):

{

"employee\_id": "number",

"engagement\_score": "number",

"teamwork\_score": "number",

"punctuality\_score": "number",

"overall\_performance\_score": "number"

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /employee/course-details/?id=

Retrieves employee details along with their course performance and project count. Allows filtering by email.

* Method: Get
* Request-Parameters: id
* Response:
  + Success Response (**200 OK**):

{

"courseStatusCount": {

"completed\_count": "number",

"failed\_count": "number",

"incomplete\_count": "number"

},

"coursesAssigned": [

{

"courseName": "string",

"courseStatus": "string",

"score": "number",

"completionDate": "string"

}

],

"topScoreAndCourse": {

"score": "number",

"courseName": "string"

},

"averageScore": "number"

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /dashboard

Retrieves an overview of course statuses by designation, courses associated with each designation, top-performing employees based on course completion and status, and top learners by completed courses.

* Method: Get
* Response:
  + Success Response (**200 OK**):

{

"topLearners": [

{

"employee\_id": "number",

"first\_name": "string",

"last\_name": "string",

"completed\_courses": "number"

}

],

"topEmployees": [

{

"employee\_id": "number",

"first\_name": "string",

"last\_name": "string",

"courseName": "string",

"score": "number"

}

],

"courseByDesignation": [

{

"designation\_type": "string",

"courseName": "string"

}

],

"courseStatusByDesignation": [

{

"designation\_type": "string",

"completed\_count": "number",

"failed\_count": "number",

"incomplete\_count": "number"

}

]

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /course/?course=

Retrieves a list of all course details, with optional search functionality based on a course name.

* Method: Get
* Request-Parameters: course
* Response:
  + Success Response (**200 OK**):

{

"courses": [

{

"course\_id": "number",

"course\_name": "string",

"course\_description": "string",

"duration\_hours": "number"

}

]

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

**Endpoint:** /course/performance/?id=

Retrieves detailed performance metrics for a specific course, including top-performing employees, department performance, designation details, and course status counts.

* Method: Get
* Request-Parameters: id
* Response:
  + Success Response (**200 OK**):

{

"overAllCoursePerformance": [

{

"employee\_id": "number",

"first\_name": "string",

"last\_name": "string",

"score": "number"

}

],

"departmentPerformance": [

{

"department": "string",

"avg\_score": "number"

}

],

"designationDetails": {

"designation\_type": "string",

"course\_name": "string"

},

"courseStatusCount": {

"completed\_count": "number",

"failed\_count": "number",

"incomplete\_count": "number"

}

}

* + Failure Response (**500 INTERNAL SERVER ERROR**):

{

"message": "Internal server error"

}

## Architecture

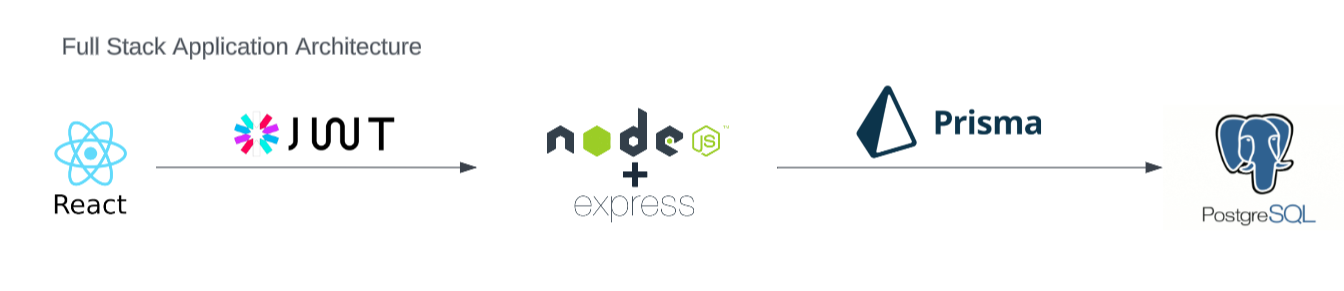
****

Figure . Full Stack Architecture Diagram

## Future Roadmap

To enhance the module's functionality and effectiveness, the following roadmap outlines the planned features and improvements:

* Enhanced Data Analytics: Develop advanced analytics capabilities to provide deeper insights into employee performance, course effectiveness, and learning paths that lead to higher retention.
* Integration with Existing Systems: Create APIs and integrations that allow seamless data exchange between this module and existing HR or learning management systems, ensuring a holistic view of employee performance and training.
* User Feedback Mechanism: Implement a system for collecting user feedback on training programs, allowing organizations to continuously refine their learning offerings based on employee input.
* Mobile Access: Develop a mobile-friendly version of the dashboards and reports, allowing administrators and employees to access performance data and learning resources on-the-go.
* Custom Reporting Tools: Introduce customizable reporting options that allow administrators to generate tailored reports based on specific metrics or employee groups.
* Real-time Notifications: Implement real-time notifications for employees regarding their training progress, upcoming deadlines, and performance feedback from managers.
* Gamification of Learning: Explore gamification strategies to increase employee engagement with training programs, making learning more interactive and rewarding.
* Expansion of Learning Content: Regularly update and expand the catalog of training courses available, ensuring that the content is relevant to the organization's evolving needs.
* Longitudinal Studies: Conduct longitudinal studies to analyze the long-term effects of training on employee performance and retention, providing valuable insights for strategic planning.

These sections provide a clear overview of your project’s purpose and outline a strategic roadmap for its future development and improvement.

# Data Engineering

## Overview

The ELT (Extract, Load, Transform) process described here is responsible for ingesting, processing, and analysing course performance data from various departments. The process extracts raw data from different sources, loads it into staging tables, transforms it through cleaning and merging, and finally creates reports and visualizations. This ELT process follows the modern data stack, where data transformations occur after loading into a central repository.

## Problem Statement

Connect and extract data from Database used for Full stack projects and use additional data (Kaggle datasets or generated data) for effectively using ETL operations.

Clean, Transform, Design and Build Data model, build reporting folder structure using Python.

## A diagram of a panda Description automatically generatedA**rchitecture**

Figure Data Engineering Architecture Diagram

**Staging Layer:**

In the staging layer, the raw CSV files are loaded, and preliminary cleaning is performed, including:

* **Type Conversion:** Converting columns to appropriate types (e.g., integer, datetime).
* **Merging:** Joining employee information with course performance, project performance, etc.

**Preparation Layer:**

This layer prepares the cleaned staging data for reporting. Key transformations include:

* **Dropping Unnecessary Columns:** Removing irrelevant columns.
* **Renaming Columns:** Standardizing column names (e.g., renaming id to employee\_id).
* **Merging Tables:** Combining different tables (e.g., merging courses and designation\_courses).

**Reporting Layer:**

In this layer, the prepared data is saved for analysis:

* **Saving to CSV:** Finalized data is saved as fact and dimension tables in the Report directory.

## Reports

* **Designation Count by Course and Status:** This report provides the count of employees per course, grouped by designation and course status.
* **Course Status Count:** A summary of courses by their completion status.
* **Top Scorer and Average Score by Designation:** Identifies the top scorer in each designation and calculates the average score per course.

## Data Warehouse Structure

The Galaxy Data Warehouse Schema, also known as a Fact Constellation Schema, acts as the next iteration of the data warehouse schema. Unlike the Star Schema and Snowflake Schema, the Galaxy Schema uses multiple fact tables connected with shared normalized dimension tables. Galaxy Schema can be thought of as star schema interlinked and completely normalized, avoiding any kind of redundancy or inconsistency of data.

Galaxy Schema Structure

* **Center of the schema:** The fact tables Course\_Performances, Project\_Performances, and

Resignation\_Records.

* **Shared Dimension:** The Employees table acts as a shared dimension across all fact tables.
* **Other Dimensions:** Courses, Designation\_Courses (bridge), and potentially Projects as other key dimensions supporting the fact tables.
* **Fact Tables**
  + **Course\_Performances**
    - **Fact Attributes:** score, completion\_date, course\_status
    - **Dimensions:** 
      * Employee (FK): Links to the Employees table.
      * Course (FK): Links to the Courses table.
  + **Project\_Performances**
    - **Fact Attributes:** engagement\_score, teamwork\_score, punctuality\_score, overall\_performance\_score
    - **Dimensions:** 
      * Employee (FK): Links to the Employees table.
      * Project (implicit dimension): If project details are maintained elsewhere, a Projects dimension table could be created.
  + **Resignation\_Records**
    - **Fact Attributes:** resignation\_date, reason
    - **Dimensions:** 
      * Employee (FK): Links to the Employees table.
* **Fact Tables**
  + **Employees**
    - **Attributes:** id, first\_name, last\_name, department, designation\_type, hire\_date, employment\_status
    - **Relationships:** Links to fact tables like Course\_Performances, Project\_Performances, and Resignation\_Records.
  + **Courses**
    - **Attributes:** id, course\_name, course\_description, duration\_hours
    - **Relationships:** Links to the Course\_Performances fact table.
  + **Designation\_Courses (Bridge Table)**
    - **Attributes:** designation\_type, course\_id
    - **Relationships:** Links DesignationType and Courses, supporting the many-to-many relationship.

## Logging

The logging method used in this ELT process is based on Python's built-in `**logging**` module, which provides a flexible framework for emitting log messages from Python programs. Logging is essential for tracking the execution flow, diagnosing issues, and understanding the transformations applied to the data throughout the pipeline.

In this process, the logging configuration is set up to capture various levels of messages, including `**INFO**`, `**ERROR**`, and `**DEBUG**`. This allows for different granularity in logging, helping distinguish between regular operational messages and error reports. The logs are written to a file, typically named `**log.txt**`, ensuring that a historical record of the process execution is maintained for review.

Key logging statements are strategically placed throughout the code, such as before and after major operations like data merging, transformations, and data saving. This not only helps in tracking progress but also in identifying the point of failure in case of exceptions. In the event of an error, detailed messages are logged, providing context about the operation being performed and the nature of the error. This systematic approach to logging facilitates easier debugging and monitoring of the ELT pipeline, ultimately enhancing its reliability and maintainability.

## Conclusion

This ELT pipeline efficiently handles course performance data, employee details, and project performance to produce meaningful reports. The modular approach with stages ensures that data is cleaned and transformed progressively, which enables the final reporting and analysis layers to deliver accurate and useful insights.

# Data Science

## Problem Statement

Promotion Forecasting: Predict which employees are most likely to be promoted based on their learning progress, skill development, and job performance metrics.

## EDA

### No. of. Unique Values

A graph of a number of unique values

Description automatically generated

Figure No. of. unique values in dataset

### Awards won on each Education category

A blue and red circle with a red triangle

Description automatically generated

Figure Awards won by education category

### Promotion counts on each Education category

A graph of blue and orange bars

Description automatically generated

Figure Promotion count on each Education category

### Gender Distribution

A red and blue rectangular bars

Description automatically generated

Figure Gender Distribution

### Length of Service by recruitment channel

A diagram of a service by recruitment channel

Description automatically generated

Figure Length of service by recruitment channel

### Age Distribution by education

A graph of different colored bars

Description automatically generated

Figure Age Distribution by education

## Data Preprocessing

Data preprocessing steps include:

* **Handling Missing Values:** Missing values in the education column are filled with the mode. The previous\_year\_rating is filled with 0 where missing.
* **Concatenation:** Training and test datasets are combined for uniform processing.
* **Duplicate Removal:** Duplicate records are removed from the dataset.
* **Interpolation:** Remaining missing values are filled using interpolation methods.

## Feature Engineering

Feature engineering focuses on creating new features that enhance model performance:

* **Correlation Analysis:** The correlation matrix is calculated to assess the relationships between features.
* **Metrics Creation:**
  + **sum\_metric:** A metric combining awards\_won?, KPIs\_met >80%, and previous\_year\_rating.
  + **total\_score:** A score calculated as the product of avg\_training\_score and no\_of\_trainings.
  + **Encoding Categorical Data:** Categorical features are encoded using LabelEncoder and one-hot encoding where appropriate. Columns like recruitment\_channel, region, and employee\_id are dropped due to redundancy or irrelevance.

## Model Training

Various classification models are trained to predict employee promotions:

* **Decision Tree:** A decision tree classifier is trained on the dataset.
* **Random Forest:** A random forest classifier is utilized for improved accuracy.
* **Logistic Regression:** A logistic regression model is also trained for comparison.

## Model Evaluation

Model evaluation involves assessing the performance of trained models using metrics like accuracy, classification reports, and confusion matrices.

|  |  |
| --- | --- |
| Model | Accuracy |
| Decision Tree | 91.09 |
| Random Forest | 94.03 |
| Logistic Regression | 93.82 |

## Conclusion

This document outlines the key steps taken in the data science project to predict employee promotions based on various factors. The EDA, feature engineering, model training, and evaluation processes provide a robust foundation for understanding employee dynamics and predicting promotions effectively. The use of multiple models allows for comparison and ensures the most accurate predictions are achieved.

**Appendix Title**

Document Title