|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| **ETL** | | | | | |
|  | | | |
|  | | | |
| Department:  **Datakulture** | | | |
| Document Name:  **PRODUCT SPECIFICATION DOCUMENT** | | | |
| Version / Date:  **0.1 / 14 Feb-2025** | | | |
|  | |  |  |

# INTRODUCTION

## DOCUMENT HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Contributors | Remarks |
| 14 Feb 2025 | 0.1 | Monisha M | Draft |
| 25 Mar 2025 | 0.2 | Monisha M | Draft |

## DISTRIBUTION LIST

|  |  |  |
| --- | --- | --- |
| Organization | Name | Signature / Date |
| Sedin Technologies | Jagadeesan.V |  |
| Sedin Technologies | Datakulture – Internal Stakeholders |  |

[0 INTRODUCTION 2](#_Toc196223455)

[0.1 DOCUMENT HISTORY 2](#_Toc196223456)

[0.2 DISTRIBUTION LIST 2](#_Toc196223457)

[**0.3** **CHANGE SUMMARY** 6](#_Toc196223458)

[0.4 ABBREVIATIONS, ACRONYMS, AND DEFINITIONS 6](#_Toc196223459)

[0.4.1 Abbreviations 6](#_Toc196223460)

[1 Process flow 7](#_Toc196223461)

[1.1 Uses 7](#_Toc196223462)

[2 Customer Registration 8](#_Toc196223463)

[2.1 Work Function: 8](#_Toc196223464)

[2.2 New Registration Popup 8](#_Toc196223465)

[2.2.1 User Input field for registration 9](#_Toc196223466)

[2.3 Registration Approval 10](#_Toc196223467)

[2.3.1 Datakulture Admin Portal’s Inbox 11](#_Toc196223468)

[2.3.2 User Input 11](#_Toc196223469)

[3 Customer Login 12](#_Toc196223470)

[3.1 Work Function: 12](#_Toc196223471)

[3.2 User Input for Login 12](#_Toc196223472)

[4 User Creation 15](#_Toc196223473)

[4.1 Work Function: 15](#_Toc196223474)

[4.2 User Input for user creation 15](#_Toc196223475)

[5 Module creation 17](#_Toc196223476)

[5.1 Work Function: 17](#_Toc196223477)

[5.2 User Input 17](#_Toc196223478)

[6 Configuration Manager 18](#_Toc196223479)

[6.1 Work Function: 18](#_Toc196223480)

[6.2 User Input 18](#_Toc196223481)

[6.2.1 Configure Connections 19](#_Toc196223482)

[7 Entity Data Mapping 21](#_Toc196223483)

[7.1 Work Function: 21](#_Toc196223484)

[7.2 User Interaction 21](#_Toc196223485)

[7.3 User Input 22](#_Toc196223486)

[7.4 Response Handling 23](#_Toc196223487)

[7.4.1 Success: 23](#_Toc196223488)

[7.4.2 Failure: 23](#_Toc196223489)

[7.5 Visual Indicators 23](#_Toc196223490)

[8 Transformation Agents 24](#_Toc196223491)

[8.1 SQL 24](#_Toc196223492)

[8.1.1 Work Function: 24](#_Toc196223493)

[8.1.2 User Input 24](#_Toc196223494)

[8.2 Python/PySpark 25](#_Toc196223495)

[8.2.1 Work Function 25](#_Toc196223496)

[8.2.2 User Input 25](#_Toc196223497)

[8.3 Deduplication/Cleaning/Fuzzy Logic 26](#_Toc196223498)

[8.3.1 Work Function 26](#_Toc196223499)

[8.3.2 User Input 26](#_Toc196223500)

[8.3.3 Open Project: 27](#_Toc196223501)

[8.3.4 Create Project: 28](#_Toc196223502)

[8.4 No Code 29](#_Toc196223503)

[8.4.1 Work Function 29](#_Toc196223504)

[8.4.2 User interaction 29](#_Toc196223505)

[8.4.3 User Input 30](#_Toc196223506)

[9 Scheduler 31](#_Toc196223507)

[9.1 Work Function 31](#_Toc196223508)

[9.2 User Input 31](#_Toc196223509)

[9.2.1 Job Scheduler Form 31](#_Toc196223510)

[9.2.2 Specific Schedule Type 32](#_Toc196223511)

[10 Trail Run - Sample Data 33](#_Toc196223512)

[10.1 Work Function 33](#_Toc196223513)

[10.2 User Input 33](#_Toc196223514)

[11 Data Governance 34](#_Toc196223515)

[11.1 Work Function: 34](#_Toc196223516)

[11.2 User Input 34](#_Toc196223517)

[11.3 User Interaction 35](#_Toc196223518)

[11.3.1 Selecting a Layer 35](#_Toc196223519)

[11.3.2 Viewing Tables and Columns 35](#_Toc196223520)

[11.3.3 Creating or Editing Rules 35](#_Toc196223521)

[11.3.4 Defining a New Rule 35](#_Toc196223522)

[11.3.5 Configuring the Rule 35](#_Toc196223523)

[11.3.6 Saving the Rule and Running Tests 35](#_Toc196223524)

[11.3.7 Executing Data Quality Checks 36](#_Toc196223525)

[11.4 Response Handling 36](#_Toc196223526)

[11.4.1 Success: 36](#_Toc196223527)

[11.4.2 Failure: 36](#_Toc196223528)

[11.5 Visual Indicators 36](#_Toc196223529)

[12 Data Security 37](#_Toc196223530)

[12.1 Work Function: 37](#_Toc196223531)

[12.2 User Input 37](#_Toc196223532)

[13 Data Lineage 39](#_Toc196223533)

[13.1 Work Function 39](#_Toc196223534)

[13.2 User Input 39](#_Toc196223535)

[13.3 User Interaction 39](#_Toc196223536)

[13.4 Visual Indicators 40](#_Toc196223537)

[14 Data Catalog 40](#_Toc196223538)

[14.1 Work Function 40](#_Toc196223539)

[14.2 User Interaction 40](#_Toc196223540)

[14.3 User Input 41](#_Toc196223541)

[14.4 Visual Indicators 41](#_Toc196223542)

[15 Time Zone Configuration 42](#_Toc196223543)

[15.1 Work Function: 42](#_Toc196223544)

[15.2 User Input 42](#_Toc196223545)

[16 Technical specification 43](#_Toc196223546)

[16.1 Tools 43](#_Toc196223547)

[16.1.1 ETL pipelines 43](#_Toc196223548)

[16.1.2 Scalability Features 44](#_Toc196223549)

[16.2 Front End 45](#_Toc196223550)

[16.3 Back End 48](#_Toc196223551)

[16.4 Scripting Languages 51](#_Toc196223552)

[16.5 Database 52](#_Toc196223553)

[16.6 Data Storage 52](#_Toc196223554)

[16.7 Data Transformation 52](#_Toc196223555)

[16.8 Data Integration Frameworks 52](#_Toc196223556)

[16.9 Other 53](#_Toc196223557)

* 1. **CHANGE SUMMARY**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ver | Change Details | Authored By | Approved By | Insert Date |
| [0.1](#_Document_change_log) | Initial issue | Monisha. M |  |  |
| 0.[2](#_Document_change_log) | Changes updated in Entity Data Mapping, Data Catalog, and Data Lineage. | Monisha. M |  |  |

## ABBREVIATIONS, ACRONYMS, AND DEFINITIONS

### Abbreviations

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

# Process flow

A diagram of a software system

AI-generated content may be incorrect.

## Uses

* **Configurable ETL (Extract, Transform, Load) processes:** The process software is open source and has plug-and-play capabilities, allowing for greater flexibility and ease of use.
* **AI Enhancement:** Capability for AI to offer table and column mapping suggestions based on the search query from the connected source.
* **Futuristic Data Warehouse**: A centralized repository that provides a structured way to consolidate, integrate, and transform data from multiple sources, including Supply Chain, Finance, Maintenance, HRMS, and other data sources.
* **BI and Analytics tools:** Users can access and analyze data from the data warehouse through these tools, which will provide features like dashboards, reports, and the capability to run ad-hoc queries.
* **Decision Making:** Organizations can leverage the BI platform to analyze data and generate actionable insights for boosting business performance. This enables efficient data analysis and reporting, providing valuable insights that contribute to improved decision-making and overall business success.
* **Data Science Capabilities:** With data science capabilities, users can create machine learning models to extract valuable insights and drive data-driven decision-making.

# Customer Registration

A registration link is provided on the Sedin website to facilitate new user sign-ups.

When users click the registration link, the system loads a pop-up screen where they can fill out all the necessary information to register. The administrator will receive a registration request for approval, and the user will receive a confirmation email upon submitting the registration request.

## Work Function:

* Provision to define the basic details of customers for registration.
* Provision to define the contact information of customers.
* Provision to submit the registration to the admin.
* Provision to accept/reject the registration by Sedin admin.

## New Registration Popup

The Customer Registration page is the initial access point for new users to enter the necessary information to set up their accounts. This page is accessed via the registration link on the Sedin website, ensuring an organized and secure collection of customer information.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/**  **Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Company/Customer Name | String | Text field | 5–20 | It shouldn’t be left blank. | The company/customer’s name cannot be empty. Please input a username. |
| Industry | String | Dropdown | 5–20 | A dropdown menu should have predefined options. If 'Other' is selected, it should allow user input.  It shouldn’t be left blank. | Field is required.  or  Error message:  Invalid data. |
| Company Website | URL | Text field | 5–20 | A website address should follow the format:  [website: name@domain.com](mailto:website:%20name@domain.com). | Field is required.  or  Error message:  Invalid website. |
| Location |  | Dropdown |  | The location should contain country, state, and city. | Field is required.  or  Error message:  Invalid data. |
| Designation/Role | String | Text Field | 5–20 |  |  |
| Full Name | String | Text field | 5-20 | The name should contain a salutation, first name, and last name. |  |
| Email | Alphanumeric | Text Field | 5-20 | An email address should follow the format:  [username@domain.com](mailto:username@domain.com). |  |
| Contact Number | Numerical | Text Field |  | The contact number should be contained with the country code.  The contact number should be verified based on the selected country. | Error message:  Invalid country code. |
| I Agree |  | Checkbox |  | On clicking, the Terms and Conditions pdf document should load. |  |

### User Input field for registration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Submit |  | Action Button |  | Mandatory fields that are highlighted should not be left blank.  The submit button will remain disabled by default until all required fields are completed. Once all mandatory fields are filled out, the submit button will be enabled. | *Highlighted are mandatory*  *Please fill in all the required fields to submit.* |
| Submit |  | Action Button |  | Successfully validated fields should be indicated with a green checkmark.  Upon clicking the submit button by the customer, the system should save the data and send the request to Sedin Admin's portal inbox. A successful message should be displayed.  The user should receive a confirmation email upon submission. | Success message:  *The request has been sent successfully to the admin. Please check your email for confirmation.”*  Error message:  Invalid data. |
| Cancel |  | Action Button |  | Upon cancelling the registration without saving the data then the system should alert a message to the user.  **YES:** If the user selected “YES” then system should close the popup without saving data.  **NO:** Back to the old state. | *Data is not saved. Do you want to discard the changes? YES NO* |

## Registration Approval

Upon submission of a registration request by a customer, it will show up in the Datakulture Admin Portal’s Inbox for the admin to review.

The Sedin administrator has the authority to approve or reject the registration. The registered user will be verified based on their provided details. The user will receive an email notification regarding the approval or rejection of their registration.

### Datakulture Admin Portal’s Inbox

The Datakulture Admin Portal’s Inbox is accessible via a side icon on the Sedin Admin page. Clicking this icon will open the Inbox page.

The Datakulture Admin Portal’s Inbox serves as the main interface for the admin team to manage and process customer registration requests efficiently. It provides a streamlined system for handling pending requests, allowing admins to approve or reject them easily.

#### Viewing Requests:

* The inbox shows a list of pending registration requests, each displaying the respective company name.
* Selecting a company name opens the detailed registration form, which includes information such as basic info, Email, Contact info, Business Domain, and additional details.

##### Approving/Rejecting Requests:

* In the detailed view, the admin clicks Approve or Reject.
* If approved, the customer account is activated, and they receive an email with login details and a portal link.
* If rejected, the customer gets a rejection email, which may include the reason for rejection.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/**  **Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Company/Customer Name |  | On Click |  | On clicking the company name in the admin’s portal inbox, it should display the registration form. | The company/customer’s name cannot be empty. Please input a username. |
| Approve |  | Action Button |  | Upon clicking the Approve button, the user account should be activated, and the user should receive an email containing their login details and a portal link. | The registration request is approved. |
| Reject |  | Action Button |  | On clicking the reject button, the user should receive a rejection email that may include the reason for the rejection. | The registration request is rejected. |

### User Input

# Customer Login

The user will receive an email from the Sedin admin containing login details and a portal link.

The Customer Login Page can be accessed by clicking this link, which is sent after their registration request is approved. Upon clicking the link, the login portal opens, allowing the customer to log in. The Customer Login Page enables customers to access their accounts using their login credentials securely and offers features for password recovery. A Forgot Password link is available below the password field.

## Work Function:

* Provision to enter the username.
* Provision to enter the password.
* Provision to have a button to login into the application.
* Provision to have an option to change the password.
* Provision to view the typed password.

## User Input for Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/**  **Value Type** | **Field Type/ Event** | **Field Length** | **Validation** | **Validation Message** |
| Username | Alphanumeric | Text Field | 5–20 | The username should contain one uppercase, lowercase, and underscore.  The username shouldn’t be left blank.  The system should authenticate the given username with the database. If not exist in the database, a validation message to be displayed.  The system should authenticate the given username with the database. If the username is “Inactive” in the database, a validation message to be displayed. | Incorrect username  *Please enter “Username*  *“Username” does not exist*  *User is inactive. please contact systems administrator* |
| Email | Alphanumeric | Text Field | 5–20 | An email address should follow the format:  [**username@domain.com**](mailto:username@domain.com)  It must be unique to ensure it doesn't duplicate any existing email addresses. | Incorrect Email. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Password | Alphanumeric | Text Field |  | The system should not allow the password field to be blank, if “Yes”, the validation message to be displayed.  Password should be encrypted before authenticating, after encryption the system should authenticate with the given username. If authentication is not successful, then systems should display a message.  On successful authentication, the system should check whether the user’s password validity date is expired or not. If expired, then the validation message to be displayed.  Information with solid fillApplicable only if password policy is defined. | *Please enter “password”*  *Please enter a valid “Password*  *Your password expired. Please contact the system administrator.* |
| Password |  | On Click (Eye Icon) |  | On clicking the password edit field, The system should display the Eye icon (Unhide/hide) button to view the entered password values. |  |
| Forgot Password |  | Hyperlink |  | On clicking the forgot password, the system should load the reset password popup with the username/Email field and confirm button.  The system should authenticate the given username/Email with the database. If not exists in the database, a validation message is to be displayed.  If exists in the database, a password reset link is sent to the registered email. | *The username does not exist, please contact the system administrator.* |
| Confirm – Forgot Password |  |  |  | On clicking the confirm button, a password registration link should be sent to the registered user email. |  |
| Reset Link |  | Hyperlink |  | On clicking the reset link in the mail, it should open the change password page in the portal to reset the password. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Login |  | Action Button |  | On successful authentication, the system should check whether the user account is active or inactive. If inactive, then the validation message to be displayed  On successful authentication, the system should check whether the user account is locked or not. If “Yes”, then the validation message is to be displayed.  Information with solid fillApplicable only if password policy is defined.  The login button will remain disabled by default until all required fields are completed.  If the login credentials are correct, the user will be redirected to the Home Page. If the credentials are incorrect, an error message will appear. | *Your account is inactive. Please contact the system administrator*  *Your account is locked. Please contact the system administrator*  Please enter the username and password.  *.*  Invalid Username or Password |

# User Creation

The User Creation screen enables administrators to efficiently create, edit, and manage user accounts within the application through an intuitive interface, offering features for both new user creation and editing existing user details.

On the Home Page, there's a Module Setup for user creation. Clicking it takes you to a page where you can create, edit, and view users.

## Work Function:

* Provision to create the username.
* Provision to edit the user.
* Provision to view the existing user.
* Provision to activate and inactive the user.

## User Input for user creation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/**  **Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| User Type |  | Dropdown |  |  |  |
| First Name | Alphanumeric | Text field | 5-20 |  |  |
| Last Name | Alphanumeric | Text Field | 5-20 |  |  |
| Email | Alphanumeric | Text Field | 5-20 | An email address should follow the format:  [username@domain.com](mailto:username@domain.com). |  |
| Username | Alphanumeric | Text Field | 5–20 | The username should contain one uppercase, lowercase, and underscore.  If the username already exists, it should display a validation message. | Exists username. |
| Password Generate |  | Action Button |  | On clicking the button, the system should auto-generate the unique password. |  |
| In Active |  | Toggle Button |  | Activate and Inactivate the user. |  |
| System Admin |  | Toggle Button |  | Enable and disable the user as system admin. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Role |  | Dropdown |  | Defines the user role. |  |
| Submit |  | Action Button |  | Mandatory fields that are highlighted should not be left blank.  The submit button will remain disabled by default until all required fields are completed. Once all mandatory fields are filled out, the submit button will be enabled.  The user should receive a validation message upon creating the user. The created user should load in the grid to edit and view. | *Highlighted are mandatory*  *Please fill in all the required fields to submit.*  *User created successfully.* |

# Module creation

Module Creation provides users with the ability to create and manage modules.

It offers administrators an efficient way to handle modules within the application through an intuitive interface, supporting both the creation of new modules and the editing of existing ones. On the Home Page, Module Creation is available for module creation. By clicking it, users are directed to a page where they can create, edit, and view modules.

## Work Function:

* Provision to create the modules.
* Provision to edit the modules.
* Provision to view the existing modules.
* Provision to activate and inactive the modules
* Provision to define task control for modules.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Module Name | String |  |  | Captures the module name (eg: Sales, Finance) |  |
| Activate | Toggle Button |  |  | Activate and inactivate the module. |  |
| Save | Action Button |  |  | Same saving logic as before |  |

# Configuration Manager

The Configuration is used to connect the source data and destination data. On the Home Page, there's a Module Setup for the configuration manager. Clicking it takes you to a page where you connect the source and destination data.

## Work Function:

* Provision to connect the source and destination data.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Source/Destination |  | Radio button |  | On clicking, the Server type should load automatically to select.  Mandatory selection between Source or Destination.  Toggles the visibility of On-Premises/Cloud |  |
| Server Type  (On-Premise/  Cloud) |  |  |  | On clicking the Server type, the Connection name and type should load automatically to select.  Mandatory selection between On-Premises/Cloud  Toggles the visibility of Connection Name, Connection Type. |  |
| Connection Name | Alphanumeric | Text Field |  | It should be Unique and mandatory.  (eg: MySql\_OrderDB\_source)  Stores the connection name in the database and displays a success message. | The connection name is saved successfully. |

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Connection Type |  | Dropdown |  | On selecting connection type, respective field should load.   * **MS SQL Connection Type:** The form will display the following mandatory fields: Hostname/IP Address, Username, Password. * **API Connection Type**: The form will display the following mandatory fields: Relative URL (Text field), Authentication Type (Dropdown: options like Basic, OAuth) * **Flat File Connection Type**: The form will display the following mandatory fields: File Path (Text field), File Type (Dropdown: options like CSV, Excel)   (MS SQL, PostgreSql, API, Flat File). |  |

### Configure Connections

#### SQL Connection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Hostname/IP Address | Alphanumeric |  |  | Mandatory, Valid IP format or hostname. |  |
| Username | Alphanumeric |  |  | Mandatory |  |
| Password | Alphanumeric |  |  | Mandatory |  |
| Save |  |  |  | * Mandatory fields that are highlighted should not be left blank. * The submit button will remain disabled by default until all required fields are completed. Once all mandatory fields are filled out, the submit button will be enabled. * The user should receive a validation message upon creating the user. The created user should load in the grid to edit and view. | Please fill in all the required fields to submit.  User is created successfully. |

#### API Connection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Relative URL | Alphanumeric |  |  | Mandatory, Valid URL Format.  Captures the stores the endpoint API. | . |
| Authentication Type | Alphanumeric |  |  | Dropdown with predefined values (Basic, OAuth)  Captures and stores the data. |  |
| Save |  |  |  | Mandatory fields that are highlighted should not be left blank.  The submit button will remain disabled by default until all required fields are completed. Once all mandatory fields are filled out, the submit button will be enabled.  The user should receive a validation message upon creating the user. The created user should load in the grid to edit and view. | Highlighted are mandatory  Please fill in all the required fields to submit.  User created successfully. |

#### Flat File Connection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| File Path | String |  |  | Mandatory, Valid path format. (eg: home/desktop/report)  Captures and stores the location of the file. |  |
| File Type |  |  |  | Dropdown with predefined values. (eg: CSV, Excel) |  |

# Entity Data Mapping

The Entity Mapping process identifies and connects relevant data entities (tables) according to business requirements, ensuring structured and meaningful data for analysis. The ETL product enables users to define table and column relationships based on business logic. AI assists by suggesting relevant tables and essential columns for specific analytical needs. Users can then review and refine these selections to ensure only business-relevant data is processed in the ETL pipeline.

From the Home Page, go to the side navigation bar and find the Data Operations icon. Clicking this icon displays a card layout with multiple options. Select the Entity Data Mapping option from this layout. Once selected, the Entity Data Mapping Page opens, providing access to its features and functionalities.

## Work Function:

* Provision to search the specific tables, business requirements Eg: sales.
* Capability for AI to offer table and column mapping suggestions based on the search query from the connected source.
* Capability to drag and drop the tables and columns between essential and non-essential sections.
* Capability to show the relevant search keywords on the side to assist users.
* Provision for users to access a dropdown to choose workflow options. (source to sliver/bronze)
* Users have the provision to select columns for mapping and defining data transformations. On the Workflow page, they can configure data movement from Source to Bronze to Silver layers using drag-and-drop mapping.

## User Interaction

**Table Mapping:**

* User clicks the search button on the Entity Mapping page.
* Search bar allows users to type business requirements (e.g., “Sales”).
* AI displays suggested tables below the search bar based on the entered requirement.
* Relevant search keywords are shown on the side to assist users.
* Essential Tables (Green): Critical tables suggested by AI.
* Inessential Tables (Grey): Less critical tables suggested by AI.
* Users can: Accept AI-suggested tables. Drag and drop tables between Essential and Inessential sections. Add tables manually from the full table list below the search bar.
* User clicks the Apply button to confirm selections.
* Popup closes, and selected tables display on the Entity Mapping page in a horizontal list with columns:
  + Table Name
  + Column Mapping
  + Workflow Mapping

**Column Mapping:**

* User clicks the Column Mapping button next to a table.
* A column selection popup opens.
* Search bar allows users to enter analysis needs (e.g., “Revenue”).
* AI suggests columns, categorized as: Essential Columns (Green): Key columns based on business logic. Inessential Columns (Grey): Less critical but potentially useful.
* Relevant search keywords appear on the side to guide users.
* Users can: Drag and drop columns between Essential and Inessential sections. Manually add custom columns.
* User clicks the Apply button to confirm selections.
* Popup closes, and selected columns display under each table on the Entity Mapping page.

**Workflow :**

* Under Workflow Mapping for each table, users access a dropdown.
* Workflow options include: Source → Silver Layer: Direct mapping. Source → Bronze → Silver: Staging and transformation.
* Selected workflow type is auto-saved.
* The chosen workflow type displays under Workflow Mapping on the Entity Mapping page.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| **Search** |  |  |  | On clicking the search button, the system should load the relevant data based on the search. |  |
| **Table Mapping** |  | **Tab** |  | * On clicking the “Table Mapping” tab, the system should load the relevant table based on the search. * Users can select the AI-suggested table or manually select a specific table. |  |
| **Column Mapping** |  | **Tab** |  | * On clicking the column mapping tab, the system should load the column data for a specific table selected in the “Table Mapping” tab. * Users can select all columns for a particular table by clicking the “All” option or manually selecting the specific columns. |  |
| **Workflow Mapping** |  | **Tab** |  | On clicking the workflow mapping, the system should allow users to select the workflow options. (source to sliver/bronze) |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Save** |  |  |  | Mapping the selected tables and columns .  Same logic |  |
| **Apply** |  |  |  | On clicking the “Apply” button, the system should map the data and mapping exported as JSON/YAML. |  |

## Response Handling

### Success:

* Mapping successfully saved.
* Data flow applied successfully.
* Mapping exported as JSON/YAML.

### Failure:

* Please select at least one table before proceeding.
* Column selection is required to apply mapping.
* Data validation failed. Please review the mappings before saving.

## Visual Indicators

* **Button States:**
  + Add Mapping Button: Always enabled.
  + Apply Buttons: Disabled until at least one selection is made.
* **Search Behavior:**
  + Suggested Keywords: Displayed on the right side of the search bar for tables and columns.
  + AI Suggestions: Displayed under search results and divided into Essential and Inessential categories.
* **Drag-and-Drop Behavior:**
  + Highlight: Dragged tables or columns show a shadow border when moved.
  + Drop Zones: Essential and Inessential sections are highlighted during drag operations.
* **Table and Column Display:**
  + Table Cards: Show table names and columns horizontally.
  + Selected Columns: Appear under the table name with a structured list.
* **Workflow Display:**
  + Selected workflows (e.g., Source → Silver, Source → Bronze → Silver) appear directly under the Workflow Mapping column.

# Transformation Agents

The Transformation Agent is a key ETL component that processes and refines raw data. It uses exact matching, Nearest Neighbor algorithms, other algorithms, and fuzzy logic for deduplication. Business rules and transformations are applied using SQL, Python, or PySpark to ensure data standardization, enrichment, and quality.

Users navigate to the Home Page, open the Data Transformation and Scheduling icon, and click the Transformation Agent icon. This opens a new page with four card layouts.

* Selecting the SQL card launches the SQL Compiler Screen
* Selecting the Python or PySpark card launches the Python/PySpark environment.
* Selecting the Cleaning/Fuzzy logic card launches the screen
* Selecting the No Card launches the screen.

## SQL

### Work Function:

* Provision to enter SQL queries in the SQL Compiler.
* Provision to compile the SQL Query.
* Provision to view the connected data sources in the side navigation bar.
* Provision to add a new data source.

### User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Side Navigation bar |  |  |  | It should display information about the connected data source on the side navigation bar. |  |
| Execute (Top-right corner of the compiler) |  | Action Button |  | On clicking the Execute button, the system should compile the SQL Query.  It should display the success result. in a tabular format below the compiler and display the error message with debugging details. | Success Message:  The SQL query runs successfully.  Error Message:  The system detects an invalid SQL syntax. |
| (+) Plus – Navigation Bar |  |  |  | On clicking, it should open a dialog for adding a new data source and should dynamically appear in the side navbar |  |

## Python/PySpark

### Work Function

* Provision for the user to write code in the Python or PySpark code in the Compiler.
* Provision to display the options for managing the data sources and selecting the execution environment in the side navigation bar.
* Provision to run the code.
* Provision to add a new data source.

### User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Side Navigation bar |  |  |  | It should display details about the connected data source and select the execution environment. |  |
| (+) Plus – Navigation Bar |  |  |  | * On clicking, it should open a dialog for adding a new data source and should dynamically appear in the side navbar. * It should be located at the top of the side navbar. |  |
| Environment Type – Navigation Bar |  | Dropdown |  | * On selecting Python, the system should load the Python compiler. * On selecting PySpark, the system should load the PySpark compiler. * The Python Compiler should be loaded by default. |  |
| Run (Top-right corner of the compiler) |  | Action Button |  | On clicking the run button, the system should execute the code.  It should display the successful result. in a tabular format below the compiler and display the error message with debugging details.  A Run button is located at the top-right corner of the compiler | Success Message:  The code runs successfully.  Error Message:  The system detects an invalid syntax. |

## Deduplication/Cleaning/Fuzzy Logic

The Transformation Agent - Deduplication/Cleaning/Fuzzy Logic module enables users to clean and standardize data by applying transformation rules, clustering similar data points, and merging duplicate values.

### Work Function

* Provision to apply the transformation rules to clean and standardize data.
* Provision to merge the duplicate values.
* Provision to open the existing project and create a new project in the navigation bar.
* Capability to display a list of destination connections (tables and columns) in a tabular format.
* Provision to define the transform, cluster, and edit option in column transformation for open and create project.

### User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Side Navigation bar |  |  |  | It should display two options for the user to select:   * Open Project and * Create Project. |  |
| Open Project (Navigation bar) |  | On click |  | * On clicking the open project, the system should display the list of “destination connections” in a tabular format. * On clicking the specific destination connection, the system should load and display all tables that are relevant to the selected data source. * On clicking the specific table from the data source, it should display the columns in a tabular format along with the top 200 rows. * On clicking the “Transformation” field in the column, the system should load the “Transform, Cluster, Edit” as dropdown values. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Create Project (Navigation bar) |  |  |  | * On clicking the Create Project button, the system should load the screen with the “Get Data” field. * On clicking the Get Data field, the system should load the below value in the dropdown. * PC (For local files) * Database (For external database connections) * Upon selecting an option, the respective page opens, allowing users to connect to a table. * Once a table is selected, the same transformation and clustering options as in the "Open Project" section are available. |  |
| Merge Selected & Recluster (Create Project) |  | Action Button |  | Merges selected clusters and reclusters. |  |
| Merge Selected & Close (Create Project) |  | Action Button |  | Merges clusters and closes popup. |  |
| Apply (Create Project) |  | Action Button |  | Apply the fuzzy logic to the data that is selected. |  |
| Save |  | Action Button |  | Applies the clustering and removes duplication | Error Message:  Transformation failed. Please check the input data. |

### Open Project:

* Click Open Project from the side navigation bar.
* The screen displays a list of destination connections in a tabular format.
* Clicking on a specific destination connection shows all tables in that data source.
* Selecting a table displays its columns in a tabular format, along with the top 200 rows.
* Each column has a dropdown box near the column name for transformation options.

#### Column Transformation:

* Clicking on a column dropdown reveals Transform and Cluster & Edit options.
* Transform Option: Selecting Transform opens another dropdown with options like Trim leading & trailing white spaces, title case, uppercase, and lowercase.
* Multiple transformations can be applied repeatedly to the same column.

#### Clustering & Editing:

* Clicking Cluster & Edit, the system should open a popup with a Method dropdown containing Nearest Neighbor, Key Collision, and Fuzzy Logic.
* After selecting any one of the Nearest Neighbor, Key Collision, should display a tabular view with:
  + **Cluster Size**: Number of clusters created for duplicate values.
  + **Row Count**: Number of rows in each cluster.
  + **Values in Cluster**: Values present in the cluster (e.g., Ima S P A, Ima spa).
  + **Merge**: Checkbox to select clusters for merging.
  + **New Cell Value**: Editable field showing the first value in the cluster by default.
* Users can merge clusters by ticking the Merge checkbox.
* Users can choose from three actions:
  + **Merge Selected & Recluster**: Merges selected clusters and recalculates new clusters.
  + **Merge Selected & Close**: Merges selected clusters and closes the popup.
  + **Close**: Closes the popup without applying changes.
* After using the Nearest Neighbor and Key Collision, the Fuzzy Logic method is selected.
* The Fuzzy Logic algorithm is applied and clicking the Apply button finalizes changes and closes the popup.
* Once the cleaning and merging process is complete, the transformed data can be truncated and loaded into the silver layer.

### Create Project:

* Clicking Create Project from the side navigation bar, the system should load a pop-up with the field “Get Data From”
* On clicking **Get Data From**, it should load two options:
  + **This PC** (for local files)
  + **Database** (for external database connections).
* Selecting an option opens the respective page, allowing users to connect to a table.
* Once a table is selected, the same transformation and clustering options as in the "Open Project" section are available.
* Data processed through Create Project cannot be loaded into the silver layer.
* Instead, users can save the cleaned data by clicking the Save button and exporting it as a CSV or PDF file on the local device.

#### Visual Indicators

* **Dynamic Table Display:** Selecting a specific table dynamically updates the displayed columns and sample data.
* **Popup Windows:**
  + The Cluster & Edit popup dynamically updates based on the selected clustering method.
  + The Merge Selected & Recluster button refreshes clusters dynamically.
* **Dropdown Behavior:**
  + The Method dropdown updates the clustering options available.
  + The Transform dropdown updates based on the column selection.

## No Code

### Work Function

* Provision to search the specific tables, business requirements Eg: sales.
* Capability for AI to offer table mapping suggestions based on the search query.
* Capability to drag and drop the tables between essential and non-essential sections.
* Capability to show the relevant search tables to assist users.
* Provision for users to access a dropdown to choose columns. (Silver).

### User interaction

**Entering Business Logic:**

* The user clicks the “Get AI suggestion” button, which enables the search bar to define the business logic in the search bar text field (e.g., "Show total sales by region").

**AI-Recommended Tables Display:**

* The system generates a list of tables from the Silver Layer based on the entered logic.
* Each table is presented in a card format, showing the Table Name, Layer Name (e.g., Silver Layer), Record Count (e.g., 2.5 million), and Last Refresh Date.

**Get AI Suggestions:**

* Users can search and add tables manually via a search bar with type-ahead suggestions.
* A table browser opens, allowing selection with filters (e.g., Table Name and Object Type).

**Table Column Mapping View:**

* Clicking a table card opens the Column Mapping Panel.
* The panel categorizes columns into:
  + Essential Columns: Critical for business analysis (AI-marked)
  + Inessential Columns: The rest of the columns.
* Users can either accept all the columns, or through drag and drop can move from essential columns list to the inessential columns list and vice versa.

**Metric-Specific Aggregation View:**

* Users can specify a Metric Column (e.g., Sales Amount).
* The system suggests aggregations: Total Sales, Average Sales, Maximum Sales, and Minimum Sales
* Users can select one or more aggregations or manually define custom calculations.

**Transformation Summary Review:**

* Before applying transformations, users see a summary screen listing: Selected Tables (with counts and metadata), Chosen Columns (categorized by importance), Selected Aggregations or Metrics
* Users can edit selections directly from the summary screen.

### User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Get AI suggestions |  | Action Button |  | On click, enable the search button. |  |
| Search |  | Text Field |  | The user enters specific business requirements.  When the search button is clicked, the system should allow the AI to suggest the tables that are required for the business logic based on the search. |  |
| Table Name |  | Text Field |  | On entering the table name, it should load the relevant tables based on the search. |  |
| Object Type |  | Text Field |  | On entering the object type, it should load the relevant tables based on the search. |  |
| Run |  | Action Button |  | On clicking the run, it should validate and execute the query. | **Success**  The query is executed successfully.  **Error**  Error. |
| Create |  | Action Button |  | On clicking the create button, it should create the table. |  |
| Table Display |  | Click |  | On mouse overing the table name, it should display the fields below.   * No. of Columns * Record Count: * Last Refresh Date: |  |

# Scheduler

* The Scheduler page enables users to create, manage, and edit job schedules. It shows existing jobs in a table with details such as Schedule Name, Status, Start Time, End Time, and an Edit option.
* Users navigate to the Home Page, where they can find the Data Transformation and Scheduling icon. Clicking this icon and then the Scheduler icon opens the Scheduler page.

## Work Function

* Provision to create, manage, and edit job schedules.
* Provision to view the existing job schedules with details.
* Capability to have “Watermark” for specific schedule type when incremental load is applied.

## User Input

### Job Scheduler Form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Schedule Name | String | Text Field |  | The name should be unique and mandatory. (eg: LoadSalesData) |  |
| Schedule Type |  | Dropdown |  | **Values:**   * Generic * Specific |  |
| Generic (Schedule Type) |  |  |  | On selecting the generic, it should run automatically based on the date given in the table. |  |
| Specific (Schedule Type) |  |  |  | On selecting the specific, it should load the following field.   * Load Type (Full, Incremental) * Repeat (Daily, Weekly, Monthly) * Interval (selected repeat type) * Start Date * End Date. |  |
| Create |  | Action Button |  | On clicking the create button, it should save and add the schedule to the list. | Succes Message:  The job schedule created successfully.  Error Message:  Invalid input. |
| Discard |  | Action Button |  | On clicking discard button, it should cancel the changes and close the job schedule form. |  |
| Watermark |  |  |  | * The watermark should display only when the Specific Load type is chosen. * Applied to all incremental loads. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Plus (+) |  | Action Button |  | * On clicking, it should create the new job schedule form and should display a success/failure message. * Button remains inactive until all mandatory fields are completed. * Newly created schedules are added dynamically to the table in the Scheduler Page. | Job schedule created successfully.  Error Message:  Invalid input. Please check your details |
| Cancel |  | Action Button |  | On clicking, it should cancel changes and close the form. |  |
| Edit |  |  |  | On clicking, it should open the Job Schedule form to modify an existing schedule. |  |

### Specific Schedule Type

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Load Type |  | Dropdown |  | Values: Full load, Incremental Load | Load type in which data is uploaded. |
| Watermark (Applied to incremental load) |  | Date |  | It should be in date format. (Appears only when Load type is Incremental) |  |
| Repeat |  | Dropdown |  | Values: Daily, Weekly, Monthly.  It should be in a time interval format.  Note: As per the SQL server scheduler. |  |
| Interval |  | Integer |  | * Gets input based on the Repeat type. (Note: 30 mins) * Captures the time interval |  |
| Start Date |  | Date |  | * Mandatory * Specifies when the schedule begins. |  |
| End Date |  | Date |  | * Mandatory * Specifies when the schedule begins. |  |

# Trail Run - Sample Data

The Trial Run feature allows users to test data transfers between a source and destination using a sample CSV file. Users can upload a CSV file, validate its format, and start a trial data transfer.

From the home page, the user navigates to the Testing & Monitoring icon in the side navigation bar. Clicking this icon opens a page displaying three card layouts. Selecting the Trial Run - Sample Run card within this layout launches the Trial Run page.

## Work Function

* Provision to test data transfers between a source and destination using a sample CSV file.
* Provision to validate its format and start a trial data transfer.
* Provision to display Source Connection Details and Destination Connection Details side by side.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Upload |  | Action Button |  | * On clicking the upload button, the system should allow the user to upload the file. * It supports onlyCSVfileformats. | Success Message:  The CSV file is processed successfully and is ready for execution.  *Error Message:*  *"Upload a CSV file."* |
| Run |  | Action Button |  | * On clicking the button, the system should trigger the trial execution process. * A trial table should be created in the source that mirrors the CSV data and transferred to the destination table for validation. |  |

# Data Governance

The Data Governance Page facilitates seamless data governance by allowing users to interact with data layers, view detailed table information, and manage rules efficiently to check and improve Data Quality.

Go to the side navigation bar from the Home Page and find the Data Operations icon. Clicking this icon displays a card layout with various options. Within this layout, select the Data Governance option. Once selected, the Data Governance Page opens, providing access to its features and functionalities.

## Work Function:

* Provision to choose a data layer (Gold, Silver).
* Provision to display the total number of tables.
* Provision to view the table information.
* Capability to establish and manage rules for monitoring and enhancing data quality.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Choose Layer |  | Dropdown |  | On clicking, the system should load the “Gold Layer, Silver Layer” as dropdown value. |  |
| Total No of Tables |  | Toggle Button |  | Displays the Total Number of Tables. |  |
| Rule |  | Dropdown |  | Set rule to the table  **E.g.:** No Null Values in Column, Column Sum in Range, Data Length, Duplicate Rows, String Format Match |  |
| View Column |  | Checkbox |  | Should show all columns and can select multiple columns |  |
| **Play** (under Run Test column) |  | Action Button |  | Runs data background against all the applied rules. |  |
| **Create Rule** (present in the tabular list) |  | Action Button |  | Opens the New rule creation’s side drawer. |  |
| **Edit** (present in the tabular list) |  | Action Button |  | Opens the side drawer. |  |
| **New Rule** (present in the side panel) |  |  |  | Opens a new session inside the drawer for creating a new rule/ edit rule. |  |
| **Run Test** (inside side drawer) |  |  |  | Initially disabled, becomes enabled after selecting rule and columns. Performs validation on selected rule. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Save** |  | Action Button |  | On clicking the save button the rule will be saved and added to the existing rule list.  Rules to check data quality are applied to each column. |  |

## User Interaction

### Selecting a Layer

* The page contains a Choose Layer dropdown with two options: Silver Later, Gold Layer
* When a layer is selected, the system dynamically loads a table listing below.

### Viewing Tables and Columns

* The loaded table contains the following headings:
* Table Name – Displays the name of each table in the selected layer.
* Total Columns – Shows the total number of columns in each table.
* Rule – Displays either a Create or Edit button depending on whether rules have been defined for the table.
* Run Test – Displays a play button for executing data quality checks.

### Creating or Editing Rules

* Each row in the table has a Rule column:
* If no rule is defined, the button displays Create.
* If a rule is already created, the button displays Edit.
* Clicking either button opens a side drawer.

### Defining a New Rule

* Inside the side drawer, the following elements are displayed:
* A New Rule button at the top.
* A list of existing rules with an Edit button next to each.
* Clicking New Rule opens a new section inside the side drawer, where the user defines a rule.

### Configuring the Rule

The New Rule section contains:

* "Select Rule" Dropdown – Offers predefined rules (e.g., No Null Values in Column).
* "View Columns" Checklist Dropdown – Allows users to select multiple columns.
* Textbox – Selected column names are displayed here automatically.
* Run Test Button – Initially disabled, becomes active once all fields are selected.
* Save Button – Saves the rule configuration.

### Saving the Rule and Running Tests

* Clicking Save triggers a toaster notification: "Rule has been created."
* The side drawer closes, and the Run Test button is now available for this table.

### Executing Data Quality Checks

* Each table row has a Run Test column with a play button.
* Clicking the play button executes the validation process for the table.
* Once the test is completed, a donut chart appears in the Run Test column, displaying the validation results.

## Response Handling

### Success:

* **When Rule is Created:** A toaster notification with the message "Rule has been created" confirms the rule creation process.
* **When Rule is Edited:** The user can modify an existing rule via the Edit button next to the rule in the side panel. The updated rule will be saved and reflected in the list of rules for the table. Along with a toaster notification with the message “Rule Name” has been edited Successfully”.

### Failure:

* If any required fields are not filled or if the selected rule is not valid, appropriate error messages or validation prompts will be shown in the UI, preventing the user from saving the rule until all conditions are met.

## Visual Indicators

* **Disabled Run Test Button:**
  + The Run Test button remains dimmed and unavailable until all required fields (rule and columns) are selected in the New Rule popup.
* **Dynamic Table Update:**
  + When a layer (Silver/Gold) is selected, the table updates dynamically to display the Total Number of Tables along with Table Names and Total Columns.
  + Selecting a specific table name updates the side drawer with corresponding rules.
* **Side Drawer Visibility:**
  + The Side Drawer appears when a table name is clicked, displaying associated rules and the New Rule button.
  + The Drawer closes when the user clicks the cancel icon on the top right.
* **Toaster Notification:**
  + A Toaster Notification appears after successfully saving a rule, confirming with the message: "Rule has been created."
  + Enabled Run Test Button:
  + Once all required fields in the New Rule popup are filled, the Run Test button becomes active, indicating readiness for validation.

# Data Security

The Data Security page offers features for managing access control across various levels, ensuring the secure handling of data. Users can implement security restrictions at the Row, Column, and Object levels to control data visibility and accessibility.

Go to the side navigation bar from the Home Page and find the Data Operations icon. Clicking on this icon will display a card layout with various options. In this layout, choose the Data Security option. Clicking on it will open the Data Security Page, granting access to its features and functionalities.

## Work Function:

* Provision to manage access control across various levels.
* Capability to enforce security restrictions at the Row, Column, and Object levels to manage data visibility and accessibility.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Side Navigation |  |  |  | The side navigation bar on this page contains three options:   * Object Level Select * Row Level Select * Column Level Select |  |
| Object Level Select |  | On click |  | * On clicking, it should display the list of table names in the destination connection in a tabular format. * Each table entry includes the following options: Assign Modules button and Enable All button. |  |
| Enable All |  | Action Button |  | On clicking, it should automatically assign access to all available modules for the selected table. |  |
| Assign Module |  | Action Button |  | On clicking, it should open a popup window displaying all the created module names in a checklist format. |  |
| Select (Assign Module) |  | Checkbox |  | * Users can select one or more modules by ticking the checkbox next to each module. * If a module already has access, its checkbox will be pre-selected. * Users can also grant or remove access by checking or unchecking the box for any previously selected module. |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Apply |  | Action Button |  | On clicking, it should enforce Object Level Security on the selected table. i.e., access to that table will be given for the selected modules. |  |
| Save |  | Action Button |  | Clicking the save button will grant access to the selected modules for the chosen table. If a module is unchecked, its access will be revoked. . |  |
| Dynamic Table List |  |  |  | * Shows all tables in the destination connection. * The popup appears when assigning access to modules and disappears once the selection is applied or canceled. |  |

# Data Lineage

* On the Home Page, the side navigation bar includes an icon labeled Data Insights and Control.
* Clicking this icon opens a new page with a two-card layout.
* Selecting the Data Lineage option within this layout navigates the user to the Data Lineage page, where they access to its functionalities.

## Work Function

* Provision to explore data relationships and dependencies.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Tabel List View |  |  |  | * It should display all tables from the Destination layer in a structured tabular format. * It should allow users to search and select a specific table from the list. |  |
| Tabel Name | Label | On Click |  | * On clicking the table name, it should display the selected table and its related tables in a card-like format. * Each card includes the Table Name and a list of Column Names. * Related tables are visually connected with linking lines to illustrate their relationships. | Success Message:  Table relationship loaded successfully.  Failure Message:  No relationships found for the selected table. |
| Column Name | Label | On Click |  | * On clicking the column name, it should display its lineage, highlighting linked columns from related tables. * It should allow users to trace how data flows between columns across tables. | Success Message:  Column lineage loaded successfully.  Failure Message:  Unable to fetch column lineage. Please try again. |

## User Interaction

* **Table List View:**
  + Displays all tables from the Destination layer in a structured tabular format.
  + Users can search and select a specific table from the list.
* **Table Relationship View:**
  + Upon selecting a table, a new page opens, displaying the selected table and its related tables in a card-like format.
  + Each card includes the Table Name and a list of Column Names.
  + Related tables are visually connected with linking lines to illustrate their relationships.
* **Column Relationship Details:**
  + Clicking on any column name displays its lineage, highlighting linked columns from related tables.
  + Users can trace how data flows between columns across tables.

## Visual Indicators

* **Table Selection Highlight:** The selected table is displayed with a blue color highlight.
* **Associated Tables Highlight:** Associated tables are displayed in grey color.
* **Table Relationship Lines:** A flow line visually connects the selected table to its associated tables, representing their relationship.
* **Column Display:** Each table card displays the column names beneath the table name.
* **Column Lineage Path:** On clicking a column name, a lineage path line appears, connecting it to related columns in other tables, showing column-level relationships.

# Data Catalog

* On the Home Page, the side navigation bar includes an icon labeled Data Insights and Control.
* Clicking this icon opens a new page with a two-card layout.
* Selecting the Data Catalog option within this layout navigates the user to the Data Catalog page, where they can explore data relationships and dependencies.

## Work Function

* Provision to explore data relationships and dependencies.

## User Interaction

**Data Collection:**

* Open Data Collection: The user clicks on the Data Collection card.
* View Curated Datasets: A horizontal tabular list displays curated datasets within the project (e.g., Transformation Dataset, Aggregated\_Business\_Dataset).
* Metadata View:
  + The user clicks on any dataset name from the list.
  + A side panel opens displaying metadata such as:
  + Dataset Name
  + Creation Date
  + Data Source
  + Last Modified Date
  + Dataset Owner
  + The side panel closes when the user clicks outside the panel or selects the close button.

**Dataset:**

* Open Dataset: The user clicks on the Dataset card.
* View Project Datasets: A page displays all datasets used in the project.
* Dataset Selection:
* The user clicks on a dataset from the list.
* A side panel appears, showing metadata about the dataset (similar to Data Collection).
* Users can click a "Use in Project" button within the panel to reuse the dataset for another project.
* The panel closes automatically when the user clicks outside or clicks the discard button.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
|  |  |  |  |  |  |
| Use in Project |  | Action Button |  | The user can use the specified dataset for their project.  **Role:** Customer Admin |  |

## Visual Indicators

* **Side panel visibility:** Opens on clicking a dataset and closes on clicking outside or close button.
* **Dynamic List Update:** Newly added datasets are immediately reflected in the dataset list without refreshing the page.

# Time Zone Configuration

The Time Zone Configuration feature enables users to choose and update their preferred time zone settings, ensuring that all system timestamps match the user's selected region for better accuracy in scheduling and logging activities.

On the home page, the side navigation bar includes an icon labeled Data Insights and Control. Clicking this icon opens a new page with a two-card layout. Within this layout, selecting the Data Catalog option takes the user to the Data Catalog page, where they can explore data relationships and dependencies.

## Work Function:

* Provision to allow users to select and update their preferred time zone settings.

## User Input

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute/Field Name** | **Data/Value Type** | **Field Type** | **Field Length** | **Validation** | **Validation Message** |
| Time Zone Configuration |  | Dropdown |  | * On clicking, it should display a list of available time zones. * Once selected, the system should automatically apply the chosen time zone for timestamps across the platform. |  |
| save |  | Action Button |  | On Selecting a time zone, it should be saved and updated. | The Time Zone is saved/Updated.  Error Message:  Failed to update time zone. |

# Technical specification

## Tools

|  |  |
| --- | --- |
| Containerization: | Docker + Kubernetes (for deployment and scalability) |
| ETL Pipelines: | Apache Airflow / Prefect (for managing data workflows) |
| Data Lineage: | OpenMetadata / Apache Atlas (to track data flow) |

### ETL pipelines

|  |  |  |  |
| --- | --- | --- | --- |
| **Tool** | **Description** | **Key Features** | **Use Case** |
| AWS Glue | * A serverless data integration service that simplifies ETL pipeline development using Apache Spark. * Supports over 70 data sources. | * Serverless, * Apache Spark, * GUI interface | Scalable distributed environments |
| Azure Data Factory | * A fully managed, serverless service for hybrid data integration. * Offers over 90 built-in connectors. | * Fully managed, * code-free pipelines, * diverse connectors | Hybrid data integration |
| Google Cloud Dataflow | A unified serverless solution for stream and batch data processing. | * Automated   resource management,   * autoscaling, * Unified stream/   batch processing,   * real-time AI | Fast data processing with analytics, cost-effective |
| Integrate.io | A unified serverless solution for stream and batch data processing. | * No code, * drag-and-drop interface, * extensive connectors, * Automated resource management, * autoscaling, * real-time AI | Fast data processing with analytics, cost-effective, Low-code/no-code users |
| Matillion | A cloud-native ETL tool for major cloud data platforms. | * Cloud-native, * Extensive library of pre-built connectors, * Gen AI features | Cloud data platforms  (e.g. Snowflake or Google Big Query) |
| Apache Airflow | An open-source workflow management platform. | * Open source, * DAG-based workflows, * Custom plugins for unsupported sources. | Complex workflow management |
| Estuary Flow | A scalable streaming ETL solution with real-time CDC support. | * Real-time CDC, * Air byte integration (Access to over 200 endpoints.) | Real-time data integration and streaming applications. |

|  |  |  |  |
| --- | --- | --- | --- |
| Meltano | * An open-source platform for creating and scheduling data pipelines. * Supports over 300 data sources and targets. | * Open source, * customizable SDKs, * extensive connectors | Flexible, scalable pipelines |
| Prefect | * Managing data workflows |  |  |

### Scalability Features

|  |  |  |
| --- | --- | --- |
| **Features** | **Description** | **Importance** |
| Horizontal and Vertical Scaling | The ability to scale both horizontally (add more nodes) and vertically (increase node capacity) to handle growing data volumes. | Ensures that the ETL process can adapt to changing data demands without significant performance degradation. |
| Modular Architecture | Breaking down ETL workflows into smaller, independent components that can be easily modified, tested, and deployed. | Enhances maintainability, extensibility, and scalability by allowing components to be scaled independently. |
| Parallel Processing | The capability to process multiple tasks simultaneously across multiple nodes or cores. | Significantly improves processing speed and efficiency, allowing for faster data integration. |
| Microservice Architecture | Structuring ETL processes as a collection of microservices that communicate via APIs. | Enhances scalability, flexibility, and resilience by allowing each service to scale independently. |
| Real-Time Processing Capabilities | Support for real-time data processing using technologies like Apache Kafka or CDC (Change Data Capture). | Enables handling of streaming data and provides immediate insights, crucial for applications requiring up-to-the-minute data updates. |
| Optimization Techniques | Features like caching, indexing, partitioning, and incremental loading to optimize performance. | Reduces data latency and redundancy, ensuring efficient processing even with large datasets. |
| Cloud-Native Support | Designed to leverage cloud computing resources for scalability and cost-effectiveness. | Allows for dynamic resource allocation and automatic scaling based on workload demands. |
| Monitoring and Feedback Mechanisms | Built-in logging, auditing, and performance monitoring to identify bottlenecks and optimize workflows. | Enables proactive maintenance and continuous improvement of ETL processes. |

## Front End

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tech** | **Description** | **Key Features/Benefits** | **Use Cases** | **AI Enhancement** | **On-Premises /**  **Cloud /**  **Hybrid** |
| HTML (Hypertext Markup Language) | The foundation of all web pages. It provides the structure and content of the user interface. Think of it as the skeleton upon which the rest of the front end is built. | * Provides the structure and semantics for all UI elements. * Ensures cross-browser compatibility. * Forms the basis for accessibility. | * Structuring the overall layout of the ETL tool. * Creating forms for user input (e.g., connection details, transformation rules). * Displaying data and information. | * Limited AI integration, but can be limited AI integration, but can be used to display AI-driven insights. * It used to display AI-driven insights. | On-Premises /  Cloud /  Hybrid |
| CSS (Cascading Style Sheets) | Used for styling and visual presentation of HTML elements. It controls the look and feel of the application, including colors, fonts, layout, and responsiveness. | * Controls the visual appearance of the application. * Enables consistent styling across different browsers and devices. * Supports responsive design for various screen sizes. | * Styling the look and feel of the ETL tool. * Creating responsive layouts that adapt to different screen sizes. * Designing visual elements such as buttons, icons, and graphs. | * Limited AI integration, but can be limited AI integration, but can be used to display AI-driven insights. * It used to display AI-driven insights. | On-Premises /  Cloud /  Hybrid |
| JavaScript  (Scripting Language) | A powerful scripting language that adds interactivity and dynamic behavior to the front-end. It allows you to manipulate the DOM (Document Object Model), handle events, make API calls, and create complex UI components. | * Enables dynamic updates and interactivity without page reloads. * Allows for complex UI interactions and animations. * Supports client-side data validation and processing. * It can be used with frameworks like React, Angular, or Vue.js for more structured development. | * Implementing interactive features like drag-and-drop pipeline design, real-time data previews, and dynamic form validation. * Handling user events such as clicks, hovers, and form submissions. * Making API calls to the back end to retrieve and update data. * Building interactive visualizations of data flow and transformations. | * Can integrate AI libraries for predictive analytics or data visualization. | On-Premises /  Cloud /  Hybrid |
| JavaScript Frameworks/Libraries (React, Angular, Vue.js) | Frameworks and libraries that provide pre-built components, tools, and patterns for building complex, single-page applications (SPAs). They help organize code, manage state, and improve development efficiency. These are highly recommended for building a complex ETL tool. | * Component-based architecture for reusability and maintainability. * State management solutions for handling data flow. * Routing capabilities for navigating between different views. * Large communities and extensive documentation. | * Organizing the code into reusable components. * Managing the application's state (data) in a structured way. * Creating a modular and scalable architecture. * Building complex UI elements such as data grids, charts, and workflow editors. | * Can be used with AI libraries for real-time data processing and visualization. | On-Premises /  Cloud /  Hybrid |
| UI Component Libraries (Material UI, Ant Design, Bootstrap) | Provide pre-designed and styled UI components (buttons, forms, dialogs, etc.) that can be easily integrated into the front-end. They accelerate development and ensure a consistent look and feel. | * Ready-to-use UI components that save development time. * Consistent styling and user experience. * Responsive design support. * Often integrate seamlessly with popular JavaScript frameworks. | * Rapidly prototyping and building the user interface. * Ensuring a consistent look and feel across the application. * Creating accessible and user-friendly components. | * Can be used to display AI-driven insights in a user-friendly manner. | On-Premises /  Cloud /  Hybrid |
| State Management Libraries (Redux, Zustand, Vuex) | State management libraries help manage the application's data in a centralized and predictable way. They are particularly useful for complex applications with multiple components that need to share data. | * Centralized data storage for easy access and modification. * Predictable state updates through actions and reducers. * Improved debugging and maintainability. * It can significantly simplify data flow in a complex ETL application. | * Managing the state of the ETL pipelines, data connections, and transformation rules. * Ensuring that all components have access to the latest data. * Implementing undo/redo functionality. | * Useful for managing AI-driven data flows and updates. | On-Premises /  Cloud /  Hybrid |
| Charting Libraries (Chart.js, D3.js, Plotly) | Used for creating interactive and visually appealing charts and graphs to display data and insights. D3.js offers the most flexibility but can have a steeper learning curve. Chart.js and Plotly are easier to use for common chart types. | * Wide range of chart types and customization options. * Interactive features like zooming, panning, and tooltips. * Data visualization for a better understanding of ETL processes. | * Visualizing data lineage and transformations. * Displaying data quality metrics. Creating dashboards to monitor ETL pipeline performance. | * Essential for visualizing AI-driven data insights and predictions. | On-Premises /  Cloud /  Hybrid |
| Testing Frameworks (Jest, Mocha, Cypress) | Used for writing and running automated tests to ensure the quality and reliability of the front-end code. | * Automated testing to catch bugs early. * Improved code quality and maintainability. * Increased confidence in the application's functionality. | * Testing UI components and interactions. * Ensuring that the front-end correctly handles different data scenarios. * Validating user input and error handling. | * Can test AI-enhanced features for robustness and accuracy. | On-Premises /  Cloud /  Hybrid |
| Bundlers (Webpack, Parcel, Rollup) | These tools bundle all the front-end assets (JavaScript, CSS, images, etc.) into optimized bundles for deployment. They also provide features like code minification, tree shaking, and hot module replacement. | * Optimized bundles for faster loading times. Code minification to reduce file sizes. * Tree shaking to remove unused code. Hot module replacement for faster development. | * Preparing the front end for deployment to production. * Optimizing the application's performance. | * Essential for optimizing AI-enhanced applications for production. | On-Premises /  Cloud /  Hybrid |
| API Calls |  | Axios / React Query (for efficient data fetching and caching) |  |  |  |
| Authentication |  | Auth0 / OAuth (depending on security needs) |  |  |  |

## Back End

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tech** | **Description** | **Key Features/**  **Benefits** | **Use Cases** | **AI Enhancement** | **Cloud-Based** | **On Premise** | **Hybrid** |
| Programming Languages (Python, Java, Scala) | Languages used for building ETL logic and orchestrating data pipelines. | * Flexibility in implementation. * Extensive libraries and frameworks. * Scalability and performance (Java/Scala). * Large community support. | * Defining data transformation and validations. * Implementing data quality checks. * Orchestrating data workflows and scheduling tasks. * Build custom connectors for various data sources. | * Python excels in AI/ML due to extensive libraries (TensorFlow, PyTorch, scikit-learn). * Java & Scala are suited for large-scale data processing  and integration with existing systems. | Yes | Yes | Yes |
| Data Warehouses (Snowflake, Big Query, Redshift) | Cloud-based data warehouses for storing and analyzing large datasets. | * Scalability and performance. * Cost-effectiveness (pay-as-you-go). * Integration with other cloud services. * Simplified data management. | * Storing transformed data for analytical purposes. * Running complex queries and generating reports. * Supporting business intelligence and data visualization. | * Integrate with AI/ML services for predictive analytics, anomaly detection, and automated data cleansing. | Yes | No | Limited |
| Databases (PostgreSQL, MySQL, Oracle, ClickHouse  Database ORM: | PostgreSQL for structured data and ClickHouse for analytical queries  SQLAlchemy for Python | * Data integrity and consistency. Transactional support. * Mature ecosystem and tools. * Well-defined schema. | * Storing and managing metadata about ETL processes. * Storing configuration settings and parameters. * Acting as source or target for data extraction and loading. | * Can be used as a source or destination for AI/ML model training data. | Yes | Yes | Yes |
| Data Lakes (AWS S3, Azure Data Lake Storage, Google Cloud Storage) | Scalable and cost-effective storage for raw and processed data in various formats. | * Scalability and Cost-effectiveness. * Support for various data formats. * Enables data discovery and exploration. | * Storing raw data before transformation. * Storing intermediate and processed data. * Enabling data scientists to access and analyze data for AI/ML projects. | * Store raw data for AI/ML model training and experimentation. | Yes | Limited | Yes |
| Message Queues  (Apache Kafka, Redis, RabbitMQ for event-driven workflows) | Asynchronous messaging systems for reliable data ingestion and processing. | * High throughput and low latency. * Reliability and fault tolerance. * Decoupling of data producers and consumers. | * Ingesting streaming data from various sources. * Orchestrating asynchronous data processing tasks. * Building real-time data pipelines for AI/ML applications. | * Enable real-time data ingestion for AI/ML-powered applications. | Yes | Yes | Yes |
| Data Processing Frameworks (SQL, Spark, Flink) | Frameworks for distributed data processing and transformation. | * Scalability and performance. * Support for various data formats and sources. * Rich set of APIs for data manipulation. * Integration with AI/ML libraries. | * Performing complex data transformations at scale. * Implementing data quality checks and cleansing. * Training AI/ML models on large datasets. * Performing real-time data analysis and inference. | * Enable large-scale data processing for AI/ML model training and inference. | Yes | Yes | Yes |
| ETL Orchestration Tools  (Airflow, Prefect, Dagster) | Tools for scheduling, monitoring, and managing ETL workflows. | * Workflow automation. * Monitoring and alerting. * Dependency management. Scalability and reliability. | * Defining and executing ETL pipelines. * Scheduling and Monitoring ETL jobs. * Managing dependencies between tasks. * Alerting on failures and anomalies. | * Integrate with AI/ML services for automated anomaly detection and pipeline optimization. | Yes | Yes | Yes |
| Containerization and Orchestration (Docker, Kubernetes) | Technologies for packaging and deploying applications in containers. | * Portability and consistency. * Scalability and resource utilization. * Simplified deployment and management. | * Packaging and deploying ETL components. * Scaling ETL infrastructure based on demand. * Managing dependencies and configurations. * Ensuring consistent execution across different environments. | * Enables portable and scalable deployment of AI/ML models and ETL components. | Yes | Yes | Yes |
| AI/ML Platforms (SageMaker, Azure ML, Google AI Platform) | Cloud-based platforms for building, training, and deploying AI/ML models. | * Simplified AI/ML development. * Scalable infrastructure. * Automated model Training and deployment. * Integration with other cloud services. | * Building and Training AI/ML models for various use cases. * Deploying AI/ML models for real-time inference. * Monitoring model performance and retraining as needed. | * Provides a comprehensive suite of tools for AI/ML development and deployment. | Yes | Limited | Yes |
| Job Scheduling: |  | Celery (Python) / Airflow (for complex scheduling) |  |  |  |  |  |
| Environment  Apache spark, native sql |  | Depends on sql(sp)) |  |  |  |  |  |

## Scripting Languages

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tech** | **Description** | **Features** | **Use Case** | **AI Enhancement** |
| Python | Used for automation and custom transformations. | * Extensive libraries (Pandas, NumPy) for data manipulation, integration with various databases and APIs, dynamic typing, and clear syntax. * Supports data science and machine learning. | * Scripting ETL workflows, and data transformations. | * Integrates well with AI/ML libraries like TensorFlow and PyTorch for predictive analytics and data quality enhancements. |
| Java | Commonly used in enterprise ETL tools for robustness. | * Platform independence, strong memory management, multithreading support, rich ecosystem of libraries and frameworks. * Used for web development, can be used with Node.js for server-side scripting. | * Building complex ETL applications. | * It can be used with AI services for real-time data processing and visualization. |
| Bash | Unix shell scripting language, used for automating system tasks and workflows. | * Command-line automation, system administration, file management, workflow orchestration, script ability, and ability to chain commands and utilities. | * Automating ETL processes on Unix systems. | * Limited AI integration, primarily used for automating tasks rather than AI-driven processes. |
| Perl | Mature language with strong text processing capabilities. | * Mature language with strong text processing, regular expressions, strong string manipulation capabilities, modular design, and extensive module library (CPAN). | * Handling complex text data transformations. | * Limited AI integration but can be used for automating data processing tasks. |
| Ruby | Simplicity and readability, often used with Ruby on Rails. | * Simplicity and readability, often used with Ruby on Rails. | * Rapid prototyping and development of web-based ETL applications. Integrating with Ruby on Rails for web development. Automating data workflows using Ruby scripts. | * Can be used with AI services for data analysis, but less common than Python. |

## Database

|  |  |  |  |
| --- | --- | --- | --- |
| **Tech** | **Description** | **Features** | **Use Case** |
| Relational Databases (e.g., MySQL, PostgreSQL) | Used for storing structured data. | ACID properties, structured data storage, SQL querying, indexing, referential integrity, and concurrency control. | Data warehousing and transactional systems. |
| NoSQL Databases  (e.g., MongoDB, Cassandra) | Ideal for handling unstructured or semi-structured data. | Flexible schema, horizontal scalability, high availability, document storage, key-value pairs, and graph databases. | Big data analytics and real-time applications. |

## Data Storage

|  |  |  |  |
| --- | --- | --- | --- |
| **Tech** | **Description** | **Features** | **Use Case** |
| Data Warehouses  (e.g., Snowflake, Redshift) | Optimized for analytics and reporting. | Columnar storage, MPP architecture, SQL interface, scalability, fast query performance, and support for BI tools. | Data analytics and business intelligence. |
| Data Lakes  (e.g., Hadoop, AWS S3) | Store raw, unprocessed data for future analysis. | Scalable storage, schema-on-read, support for various data formats, cost-effective, integration with big data processing frameworks. | Big data processing and machine learning. |

## Data Transformation

|  |  |  |  |
| --- | --- | --- | --- |
| **Tech** | **Description** | **Features** | **Use Case** |
| Apache Spark | Used for large-scale data processing and transformations. | In-memory processing, distributed computing, fault tolerance, support for multiple languages, machine learning libraries, and real-time streaming. | Real-time data processing and analytics. |
| Apache Beam | Unified programming model for both batch and streaming data processing. | Unified API, portability across execution environments, support for batch and streaming, fault tolerance, flexible data pipelines. | Complex data pipelines with multiple processing modes. |

## Data Integration Frameworks

|  |  |  |  |
| --- | --- | --- | --- |
| **Tech** | **Description** | **Features** | **Use Case** |
| Apache Airflow | Used for workflow management and orchestration. | DAG-based workflow definition, scheduling, monitoring, alerting, extensibility, integration with various systems, open-source. | Managing and scheduling ETL workflows. |
| Apache NiFi | Handles data flow and process in real-time. | Flow-based programming, data provenance, visual interface, data buffering, prioritization, real-time data processing, data routing, data transformation. | Real-time data integration and processing. |

## Other

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **CPU** | **RAM** | **Storage** |
| Frontend (React) | 1 vCPU | 1GB | 5GB |
| Backend (FastAPI/Flask) | 2 vCPU | 4GB | 20GB |
| PostgreSQL | 2 vCPU | 4GB | 50GB |
| ClickHouse | 4 vCPU | 8GB | 100GB |
| Message Queue | 2 vCPU | 4GB | 10GB |
| Airflow/PySpark | 4 vCPU | 8GB | 50GB |
| Storage (S3/Blob/Cloud Storage) | - | - | 50GB |