CodeGenie 1.0.0 – Setup Guide

CodeGenie is a modular, AI-powered code conversion platform designed to simplify and accelerate migration to Microsoft Fabric and other modern data platforms. It enables users to convert existing data workflows and codebases—such as Databricks PySpark code, Hadoop workloads, database stored procedures, and Azure Data Factory (ADF) pipelines—into Fabric-compatible implementations with minimal manual effort.
Users can upload source files, create workspaces, select appropriate Blueprints, and apply predefined rules to guide the conversion process. The platform automatically generates structured, Fabric-ready code while preserving business logic, ensuring consistency, and reducing migration complexity.

You can choose to implement the:

1. **Databricks PySpark → Fabric PySpark**
   Convert Databricks PySpark code into Microsoft Fabric-compatible code for seamless migration.
2. **Stored Procedure → Fabric PySpark**
   Transforms traditional SQL stored procedures into Fabric-supported PySpark implementations.
3. **ADF Pipeline → Fabric Conversion**
   Migrates Azure Data Factory (ADF) pipelines into Microsoft Fabric pipelines while preserving workflow logic and orchestration.
4. **Hadoop (Java) → Databricks PySpark**
   Converts Hadoop Java-based workloads such as MapReduce, Hive, or Pig logic into Databricks-compatible PySpark code.
5. **Hadoop (Python) → Databricks PySpark**
   Converts Hadoop Python-based processing scripts into scalable Databricks PySpark implementations.
6. **Azure Databricks → GCP Databricks**
   Converts Azure Databricks workloads into GCP Databricks-compatible implementations while maintaining processing logic and structure.

This solution is designed to make code and pipeline migration faster, more reliable, and accessible, especially for users who are not deeply experienced with modern data platforms. By leveraging Blueprint-driven conversion and rule-based standardization, CodeGenie minimizes manual rework, preserves business logic, and generates structured, platform-compatible outputs. This enables organizations to modernize their data workflows efficiently while improving consistency, scalability, and migration speed.

Note: Dashboard testing capabilities for Power BI and Tableau are currently under evaluation and testing. These features are planned for inclusion in future releases of CodeGenie.

CodeGenie supports role-based access control: **Admin and User**. Each role defines the level of access and actions a user can perform within the platform, ensuring secure and efficient usage

PRESCIENCE
A Movate Company

Table of Contents

1. Roles in CodeGenie

Depending on the **assigned role** — either **Admin** or **User** — you will have access to different capabilities within the CodeGenie:

Admin Capabilities

Admins can:

1. Manage users through the User Management dashboard
2. Create new users with required details such as first name, last name, email, username, and password
3. Assign roles (Admin/User) during user creation
4. View user information including ID, username, GUID, status, and timestamps
5. Monitor user status and account activity
6. Update user details when required
7. Access conversion history and execution logs
8. Monitor detailed conversion history, execution metrics, token usage, and processing logs.
9. Configure Azure OpenAI and LLM-related settings
10. Modify UI and application-level settings
11. Manage LLM-as-a-Judge configuration settings

User Capabilities

Users can:

1. Access the CodeGenie dashboard based on assigned role
2. Create and manage workspaces for different conversion tasks
3. Upload, open, and manage source files such as Databricks notebooks, ADF pipelines, Hadoop files, and stored procedures
4. Select appropriate conversion types for migration and automation tasks
5. Choose and apply predefined Blueprints for conversion
6. Configure and customize Blueprints and associated rules (if permitted)
7. Execute conversion using the **Run** functionality
8. Validate converted output before saving or downloading
9. View converted Fabric-compatible code in the output panel
10. Check conversion details such as confidence score, execution status, and validation results
11. Review cell-level information and conversion summaries
12. Save converted outputs within the workspace
13. Download the generated Fabric-compatible code
14. Manage and organize files within the workspace environment
15. Re-run or refine conversions by updating Blueprints or rules

2. Getting Started

Below are the steps to get started with and use the CodeGenie application.

Login using the vadmin

Initial login is done using the vadmin account to configure basic settings. After setup, Admin and User accounts can be created and assigned access.

The details for logging into the CodeGenie tool are as follows:
Application URL:
Default Username: vadmin
Default Password: Vadmin321$

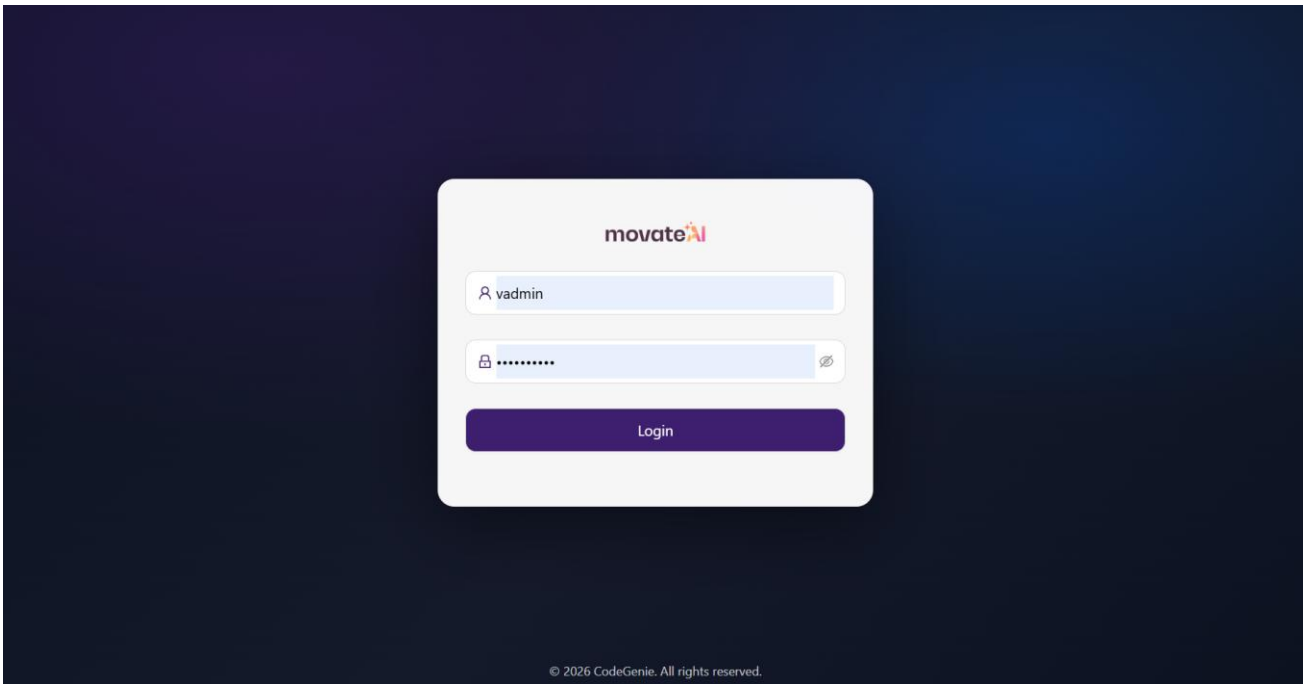

Image 1: Login page

Change Password

After logging in using the default vadmin credentials, it is strongly recommended to update the password to ensure account security.

Follow the steps below to change the password:

1. **Click on the user icon** (top-right corner of the page)
2. From the dropdown menu, select **Change Password**

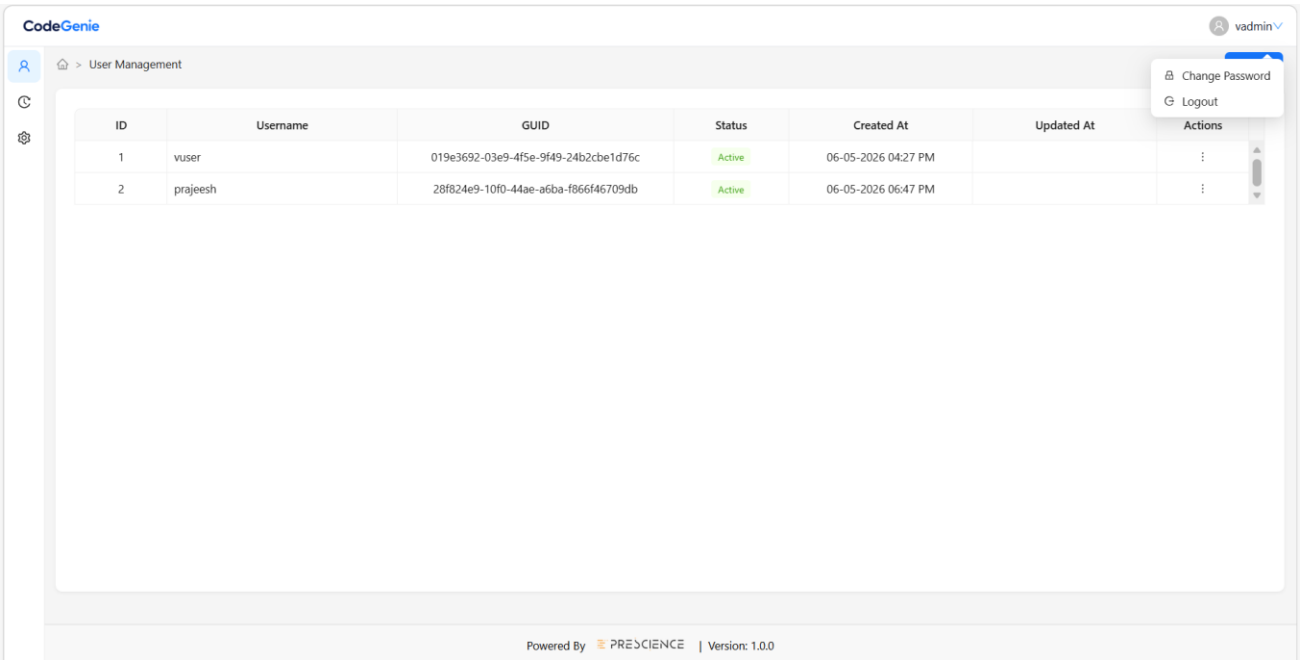

Image 2: Accessing Change Password from user menu

3. A "Change Password" dialog box will appear
4. In the **New Password** field, enter your new password
5. In the **Confirm Password** field, re-enter the new password to confirm
6. Once both fields are filled correctly, click on the **Update** button

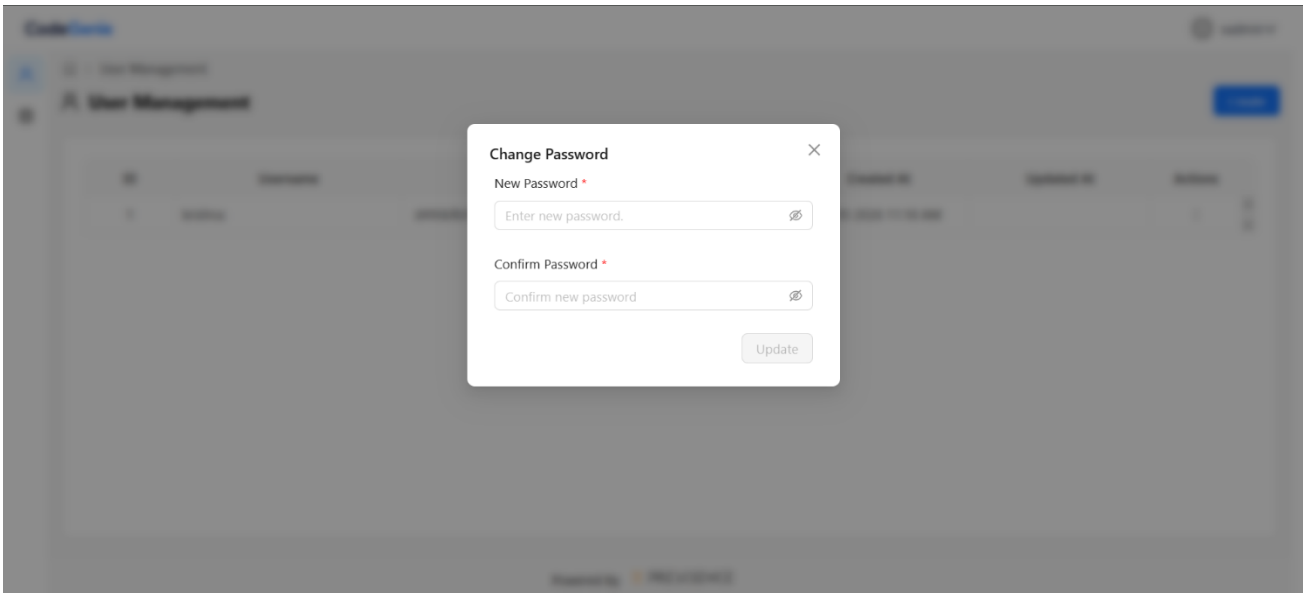

Image 3: Change Password dialog box

Update Company logo

To set up your company logo to display on the login page, follow these steps:

1. From the sidebar, click on the gear icon (⚙) to open the Settings section
2. Locate the UI Settings (JSON) Field

3. Update the login_logo_url Attribute
4. Replace the current URL with the new logo URL
5. login_logo_background_color → Set background color
6. login_button_color → Customize button color
7. navbar_background_color → Set navbar color
8. Next to the UI settings JSON box, click Save to apply the changes
9. After logout, your company logo will appear on the login page

Note: Ensure the image URL used for login_logo_url is publicly accessible and in .png or .jpg format for proper rendering

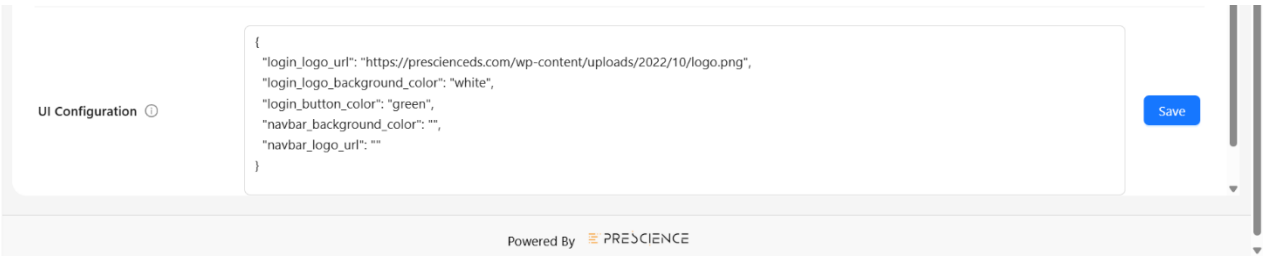

Image 4: Settings page to update company logo

3.User Management

The User Management section allows administrator to create and manage user accounts with specific roles such as “Admin” or “User”. This setup ensures secure access control for team members based on their responsibilities, such as analytics, business use, or IT operations.

Create a User

Go to the sidebar and click the **User icon** to open the **User Management** page. Here, you can view existing users along with their Username, GUID, Role Tags, Status, and timestamps.

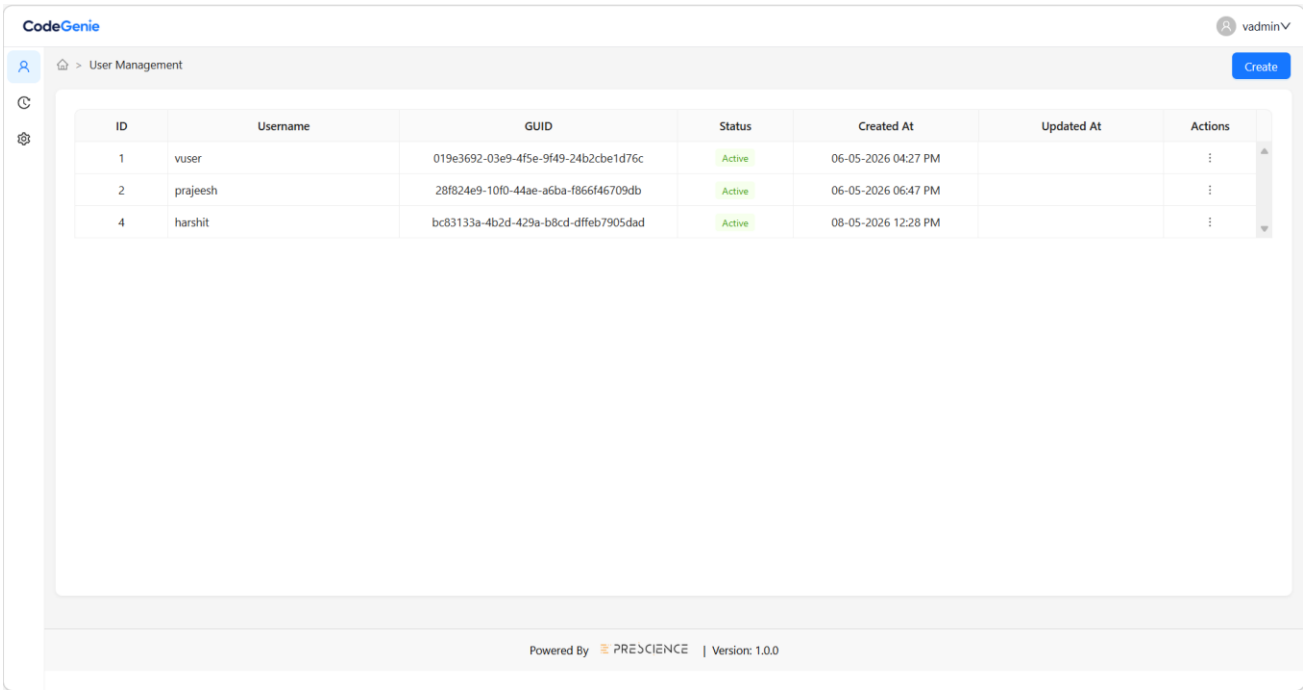

Image 5: Navigate to User Management page

Click the blue **"Create"** button on the top right of the User Management page.

Fill in the following required fields:

- **First Name** (e.g., *Prescience Decision Solutions*)
- **Last Name** (e.g., *Pvt. Ltd.*)
- **Email ID** (e.g., *info@presienceds.com*)
- **Username** (e.g., *prescience*)
- **Password & Confirm Password**
- **Assign Role** (select either *Admin* or *User*)

Click **Save** once all fields are filled.

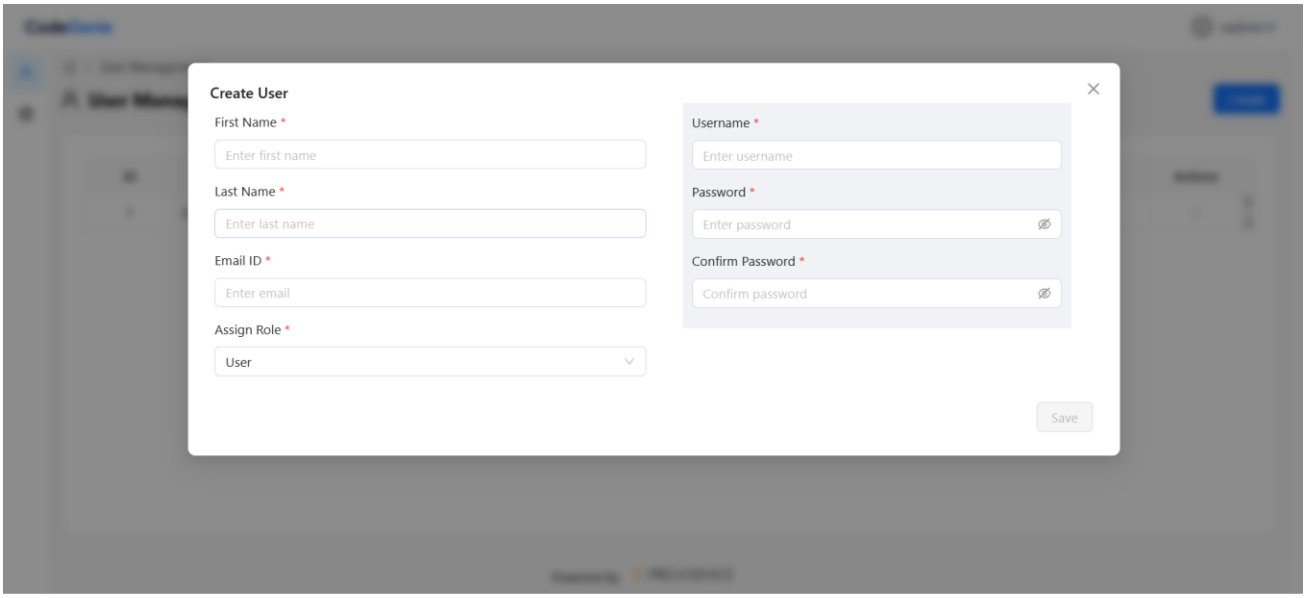

Image 6: Enter user details and assign role

A green success notification will appear on the top right confirming the user was created. The new user will now appear in the user list with their GUID, assigned tags, and status as “Active.”

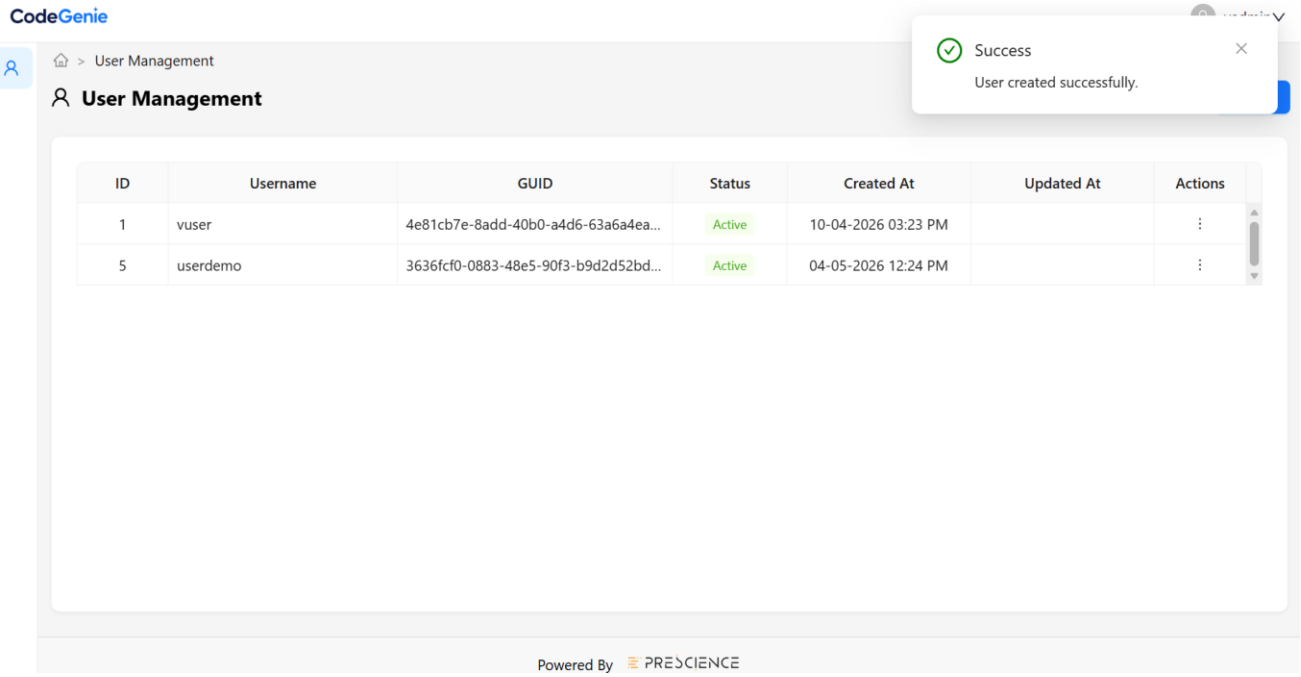

Image 7: User created successfully

4.LLM Configuration

The **LLM Configuration** section allows Admin users to configure the Azure OpenAI settings used for AI-powered code conversion and validation within the platform.
Admins can manage configuration parameters such as:

- Azure OpenAI API Key
- Azure OpenAI Endpoint
- API Version
- Deployment Model Name

The platform also provides:

- AI Confidence Threshold configuration
- LLM-as-a-Judge configuration settings for evaluation workflows

Administrators can update these JSON-based configurations directly from the Settings page and save the changes for immediate use within the application.

Updating LLM Configuration

Follow these steps to create and configure an LLM for your CodeGenie instance:

Click the **Settings icon** in the sidebar. This opens the LLM Configurations page, where existing configurations will be listed.

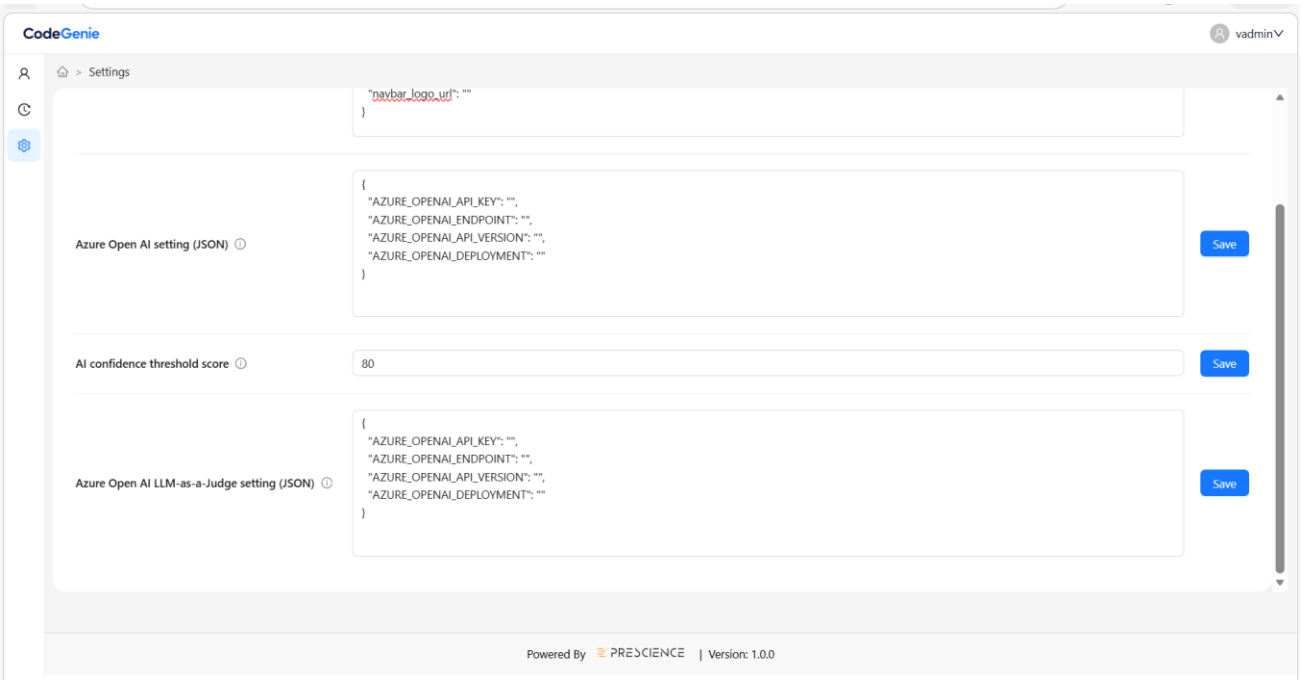

Image 8: LLM Configurations page – No existing LLMs

Fill out the form:

Navigate to the Settings page and locate the **Azure Open AI setting (JSON)** and **Azure Open AI LLM-as-a-Judge setting (JSON)** section.

Update the configuration JSON with the following fields:
{
"AZURE_OPENAI_API_KEY": "<Your API Key>",
"AZURE_OPENAI_ENDPOINT": "<Your Endpoint>",
"AZURE_OPENAI_API_VERSION": "<API Version>",
"AZURE_OPENAI_DEPLOYMENT": "<Deployment Model Name>"
}

After entering the required configuration values, click the Save button to apply the settings.

You can also configure:

- **AI confidence threshold score**

5.Logging in as a User

Once the user account is created (as described in the User Management section), users can log into the CodeGenie platform using their credentials. The platform allows users to create and manage workspaces, select blueprints for different migration types, and perform code or pipeline conversions through an interactive interface.

Users can view source and converted code side-by-side, run and validate conversions, save progress, and download generated output files. CodeGenie supports multiple migration workflows such as ADF Pipeline to Fabric Pipeline, Azure Databricks to GCP Databricks, and Hadoop to Databricks conversions while preserving business logic and maintaining consistency.

Access the User Dashboard

Follow these steps to log in as a User:

Enter the User role login credentials (e.g., Username: userdemo, Password: ******) that you had created in the User Management section.

Click the **Login** button.

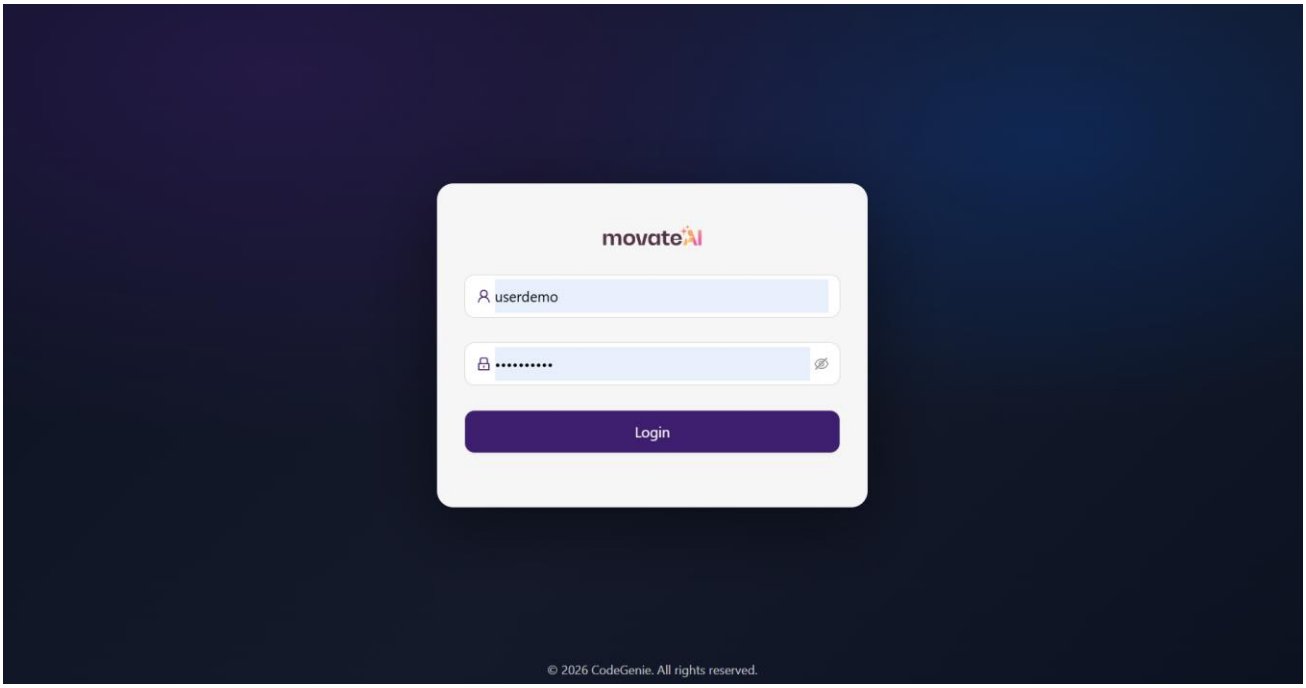

Image 9: User login screen

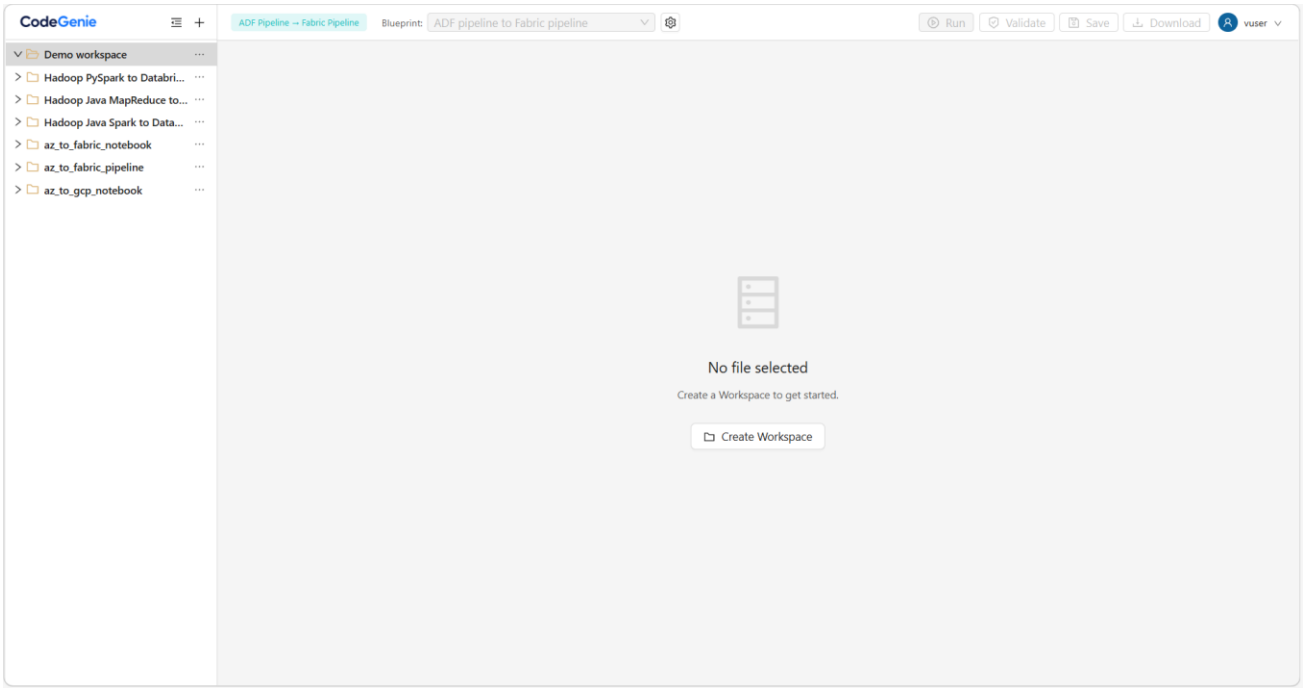

Image 10: User dashboard screen

The User Dashboard is the main workspace in CodeGenie where users can manage workspaces, files, and conversion tasks. The left panel provides a file explorer to access different migration projects and source files. The central workspace displays the selected file or shows a default screen when no file is selected. Users can create a workspace, choose a blueprint, and start working on conversion tasks.

The top bar allows users to manage blueprints and perform actions like Run, Validate, Save, and Download. The interface also provides a side-by-side view of source and converted output, enabling users to efficiently perform code and pipeline conversions.

6. Workspace Creation & Configuration

Creating a Workspace

Users begin by clicking on the “Create Workspace” button available on the dashboard. This opens a configuration dialog where users define the workspace details required for code or pipeline conversion. In this step, users are required to:

- Enter a **Workspace Name** eg: Sales Pipeline Q4
- Select a **Conversion Type**
- Choose an appropriate **Blueprint** based on the migration use case
- Optionally create a new Blueprint using **“+ Blueprint”** for custom logic and rules
- Optionally enable Code Validation before creating the workspace

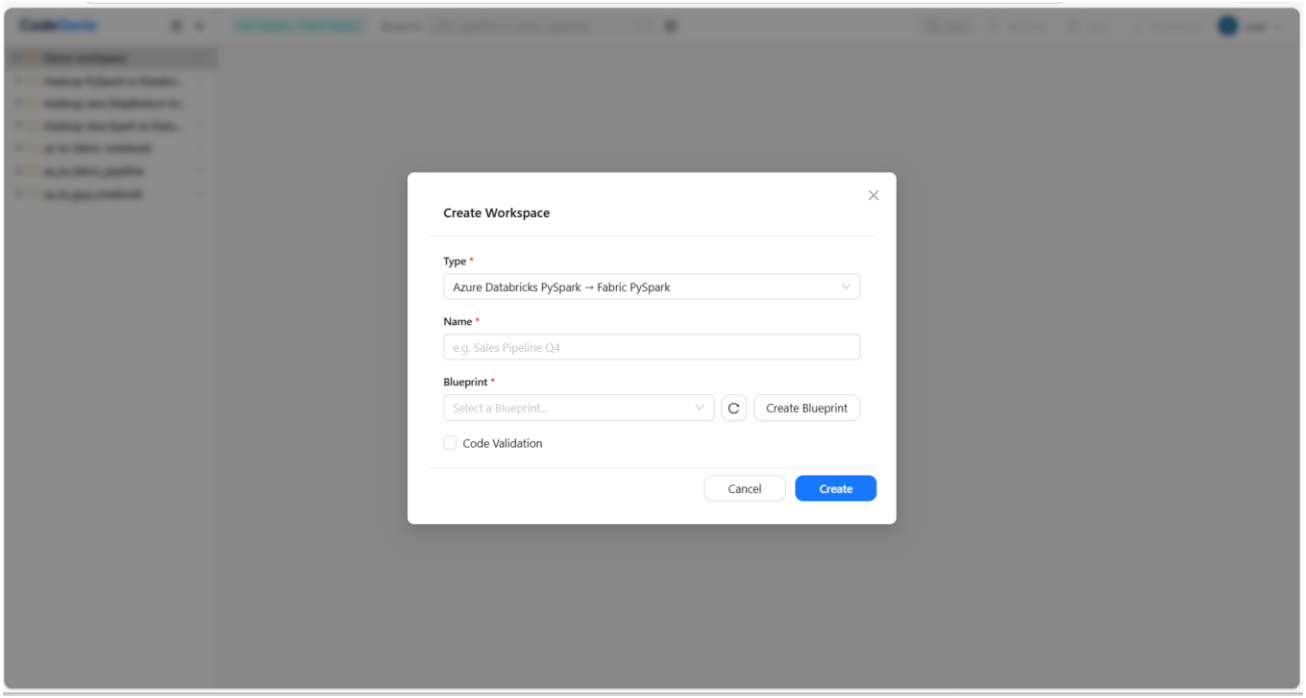

Image 11: Add workspace dialogue box

Selecting Conversion Type

The **Conversion Type** defines the type of transformation or automation the user wants to perform.

Available Conversion Types:

- **Azure Databricks PySpark → Fabric PySpark**
  Converts Azure Databricks PySpark notebooks and ETL code into Microsoft Fabric-compatible PySpark implementations for migration to Fabric environments.

- **ADF Pipeline → Fabric Pipeline**
  Migrates Azure Data Factory (ADF) pipelines into Microsoft Fabric pipelines while preserving workflow structure, activities, and business logic.

- **Stored Procedure → Fabric PySpark**
  Converts traditional SQL stored procedures into Fabric-compatible PySpark implementations for modern data engineering workflows.

- **Hadoop → Azure Databricks PySpark**
  Transforms Hadoop-based workloads such as MapReduce, Hive, or legacy Spark jobs into Azure Databricks PySpark code for cloud modernization.

- **Azure Databricks PySpark → GCP Databricks PySpark**
  Converts Azure Databricks PySpark workloads into Google Cloud Databricks-compatible implementations while adapting platform-specific configurations.

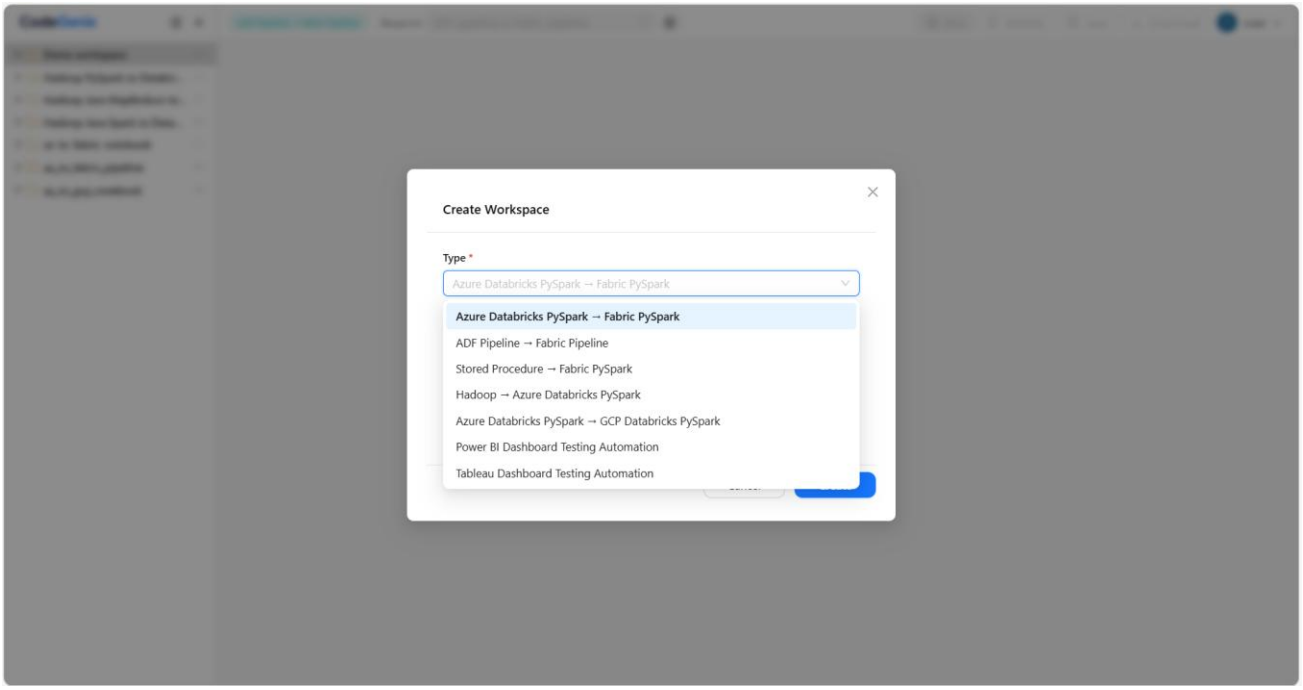

Image 11: Select conversion type

Press **Enter** or click the send icon (➤) next to the input box.

Selecting Dashboard Type (For Test Automation Only)

When users select either Power BI Dashboard Testing Automation or Tableau Dashboard Testing Automation as the conversion type, the platform configures the workflow based on the selected dashboard technology.

Available Dashboard Types

- **Power BI Dashboard Testing Automation** → Used for automated testing and validation of Power BI dashboards
- **Tableau Dashboard Testing Automation** → Used for automated testing and validation of Tableau dashboards

This ensures that test cases and validation workflows are generated according to the selected BI platform.

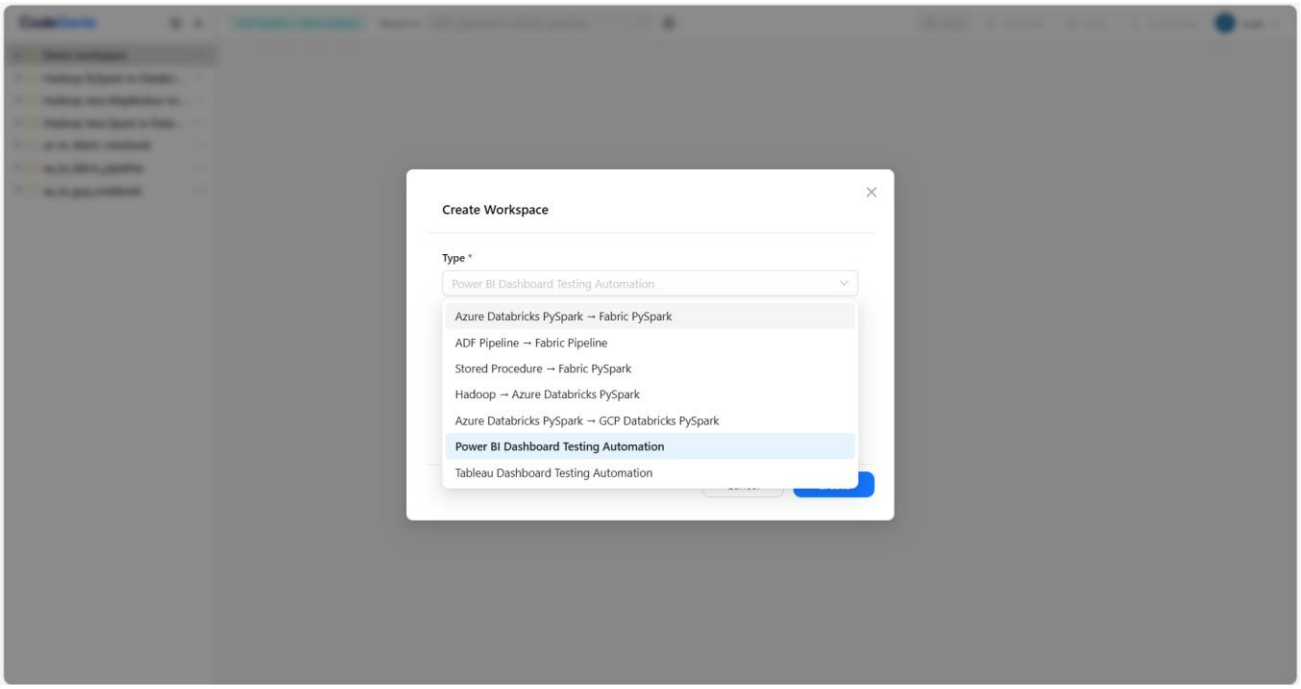

Image 11: Select dashboard automation

Selecting Blueprint

Blueprints define the conversion logic, rules, and workflow used by the system during migration or automation. Users can select from predefined blueprints based on their migration use case.

Available Blueprints

- **ADF pipeline to Fabric pipeline**
  Converts Azure Data Factory pipelines into Microsoft Fabric pipelines while preserving activities, workflow logic, and dependencies.

- **Azure Databricks to GCP Databricks**
  Converts Azure Databricks PySpark workloads into Google Cloud Databricks-compatible implementations with required platform adaptations.

- **Databricks PySpark to Fabric PySpark**
  Transforms Databricks PySpark notebooks and ETL code into Microsoft Fabric-compatible PySpark implementations.

- **Hadoop (Java) To Databricks PySpark**
  Converts Hadoop Java-based workloads such as MapReduce jobs into Databricks PySpark code for cloud modernization.

- **Hadoop (Python) To Databricks PySpark**
  Migrates Hadoop Python-based processing scripts into Databricks PySpark implementations while preserving processing logic.

- **Stored Procedure to Fabric PySpark**
  Converts traditional SQL stored procedures into Microsoft Fabric-compatible PySpark implementations.

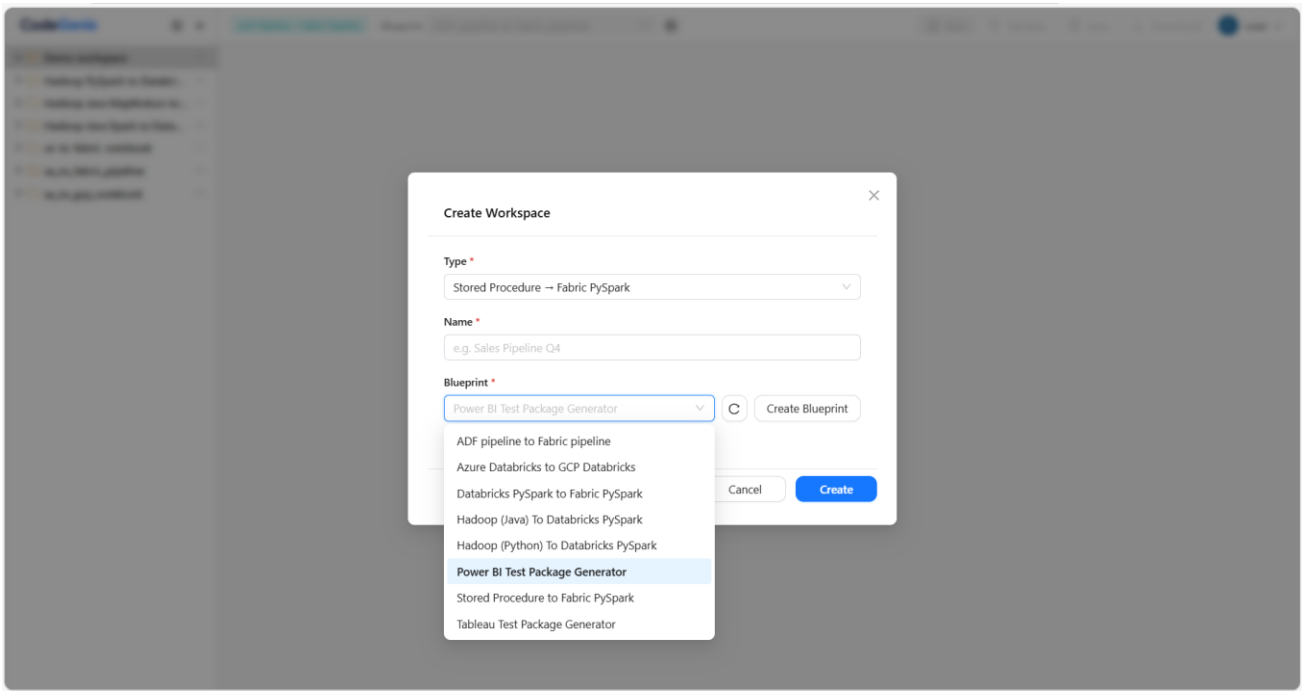

Image 12: Select Blueprint

Creating Custom Blueprint (Optional)

Users can click on **"Create Blueprint"** to define custom blueprints and configure their own conversion rules and workflows. This provides flexibility for handling specific business logic, coding standards, migration patterns, and transformation requirements based on project needs.

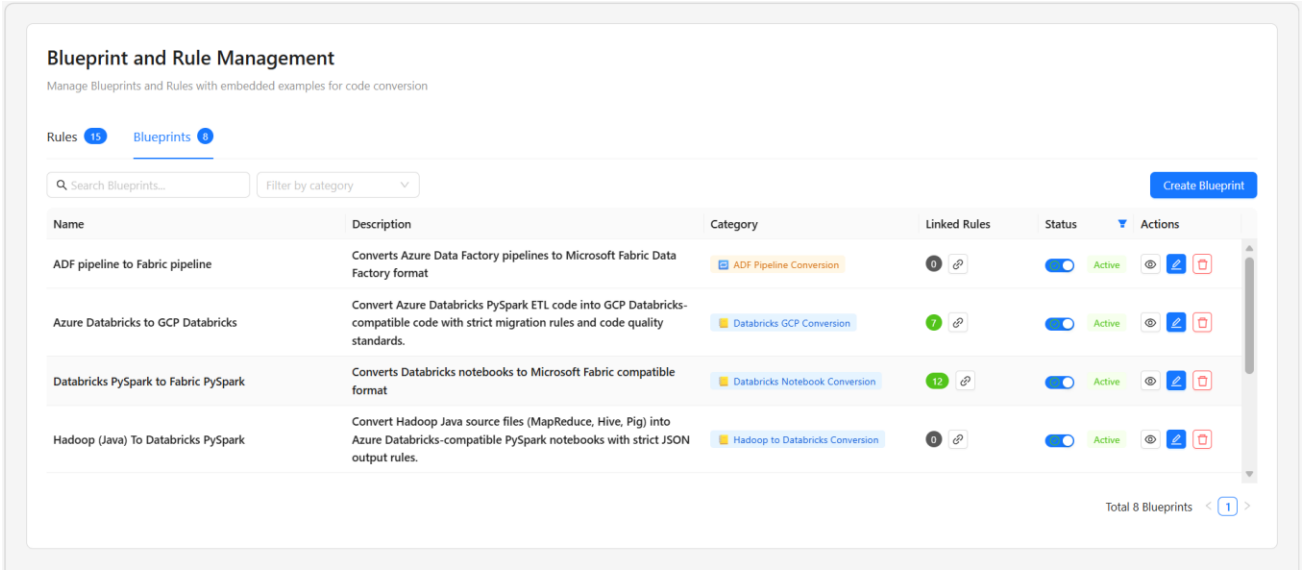

Image 13: Blueprint and Rule Management

7. Blueprint & Rule Management (Custom Blueprint Creation)

Create Blueprint

User clicks on **"Create Blueprint"** and fills:

- **Name** – Unique name of the Blueprint
- **Description** – Short explanation of the Blueprint

- **Category** – Select conversion category
- **Status Toggle** – Enable/disable the Blueprint

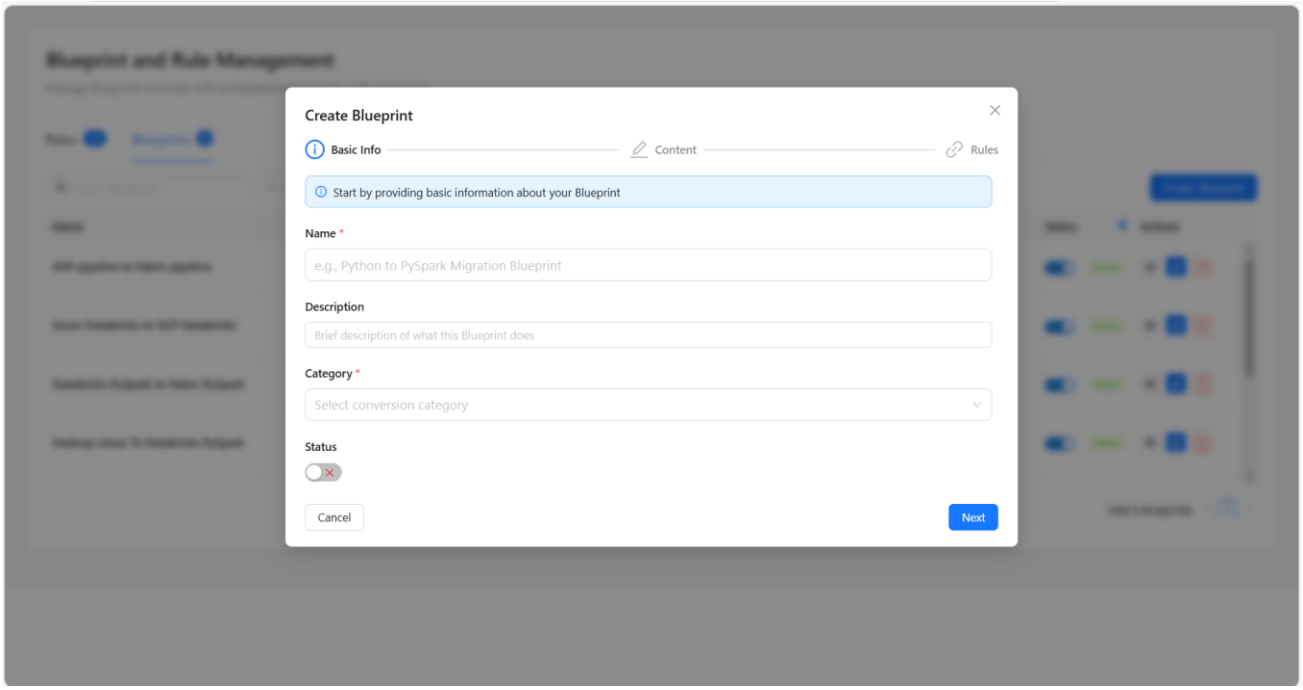

Image 14: Create Blueprint dialogue box

Blueprint Categories

• **Databricks Notebook Conversion –** Converts Databricks notebooks and PySpark workloads into Microsoft Fabric-compatible implementations.

• **Databricks GCP Conversion –** Transforms Azure Databricks PySpark workloads into Google Cloud Databricks-compatible code and configurations.

• **Hadoop to Databricks Conversion –** Converts Hadoop-based processing workloads into Databricks PySpark implementations for cloud modernization.

• **SQL/Stored Procedures Conversion –** Transforms SQL stored procedures into Fabric-compatible PySpark implementations while preserving business logic.

• **ADF Pipeline Conversion –** Converts Azure Data Factory pipelines into Microsoft Fabric pipelines while maintaining workflow structure and dependencies.

• **Hadoop to Azure Databricks PySpark –** Migrates Hadoop Java or Python workloads into Azure Databricks PySpark code for scalable cloud processing.

After filling in the required details, click on the “Next” button to continue.

Blueprint Content

User provides **Blueprint Content (System Blueprint)**

Description

- Defines the base instructions for code conversion.
- Written in natural language but acts as a system-level guideline.
- Ensures consistency, logic preservation, and structured output.

After filling Blueprint content, click on **Next button**

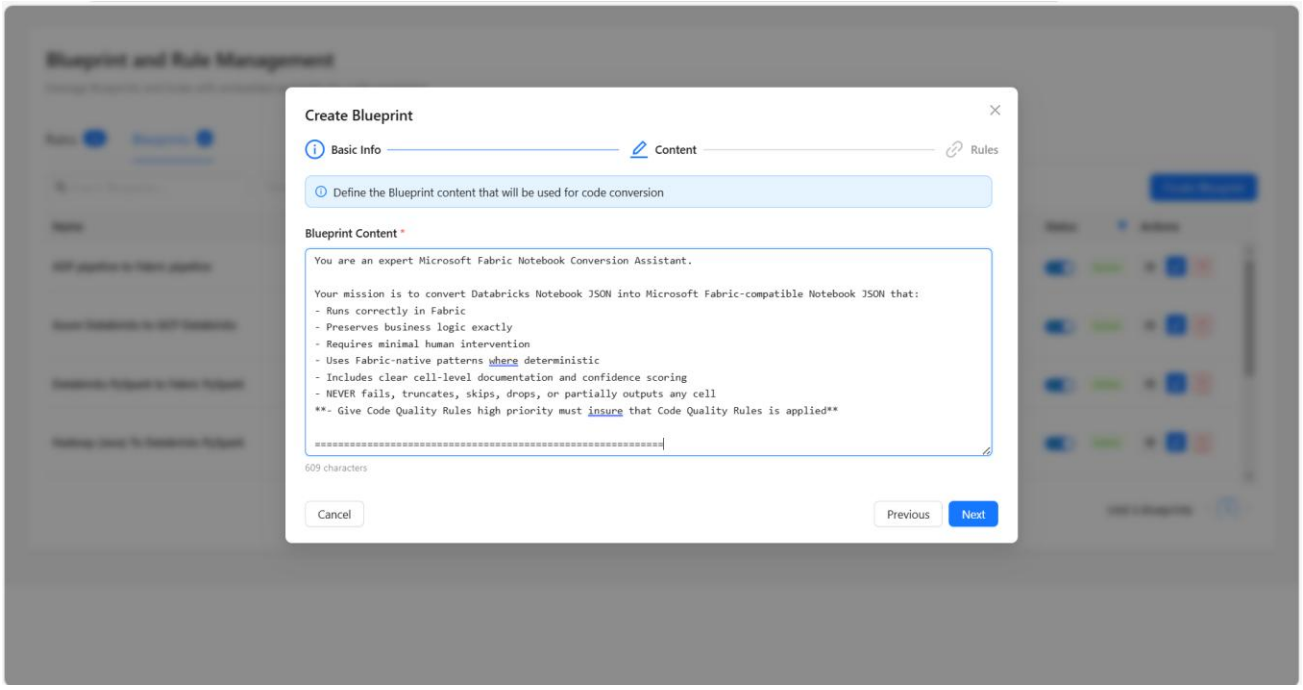

Image 15: Create Blueprint content dialogue box

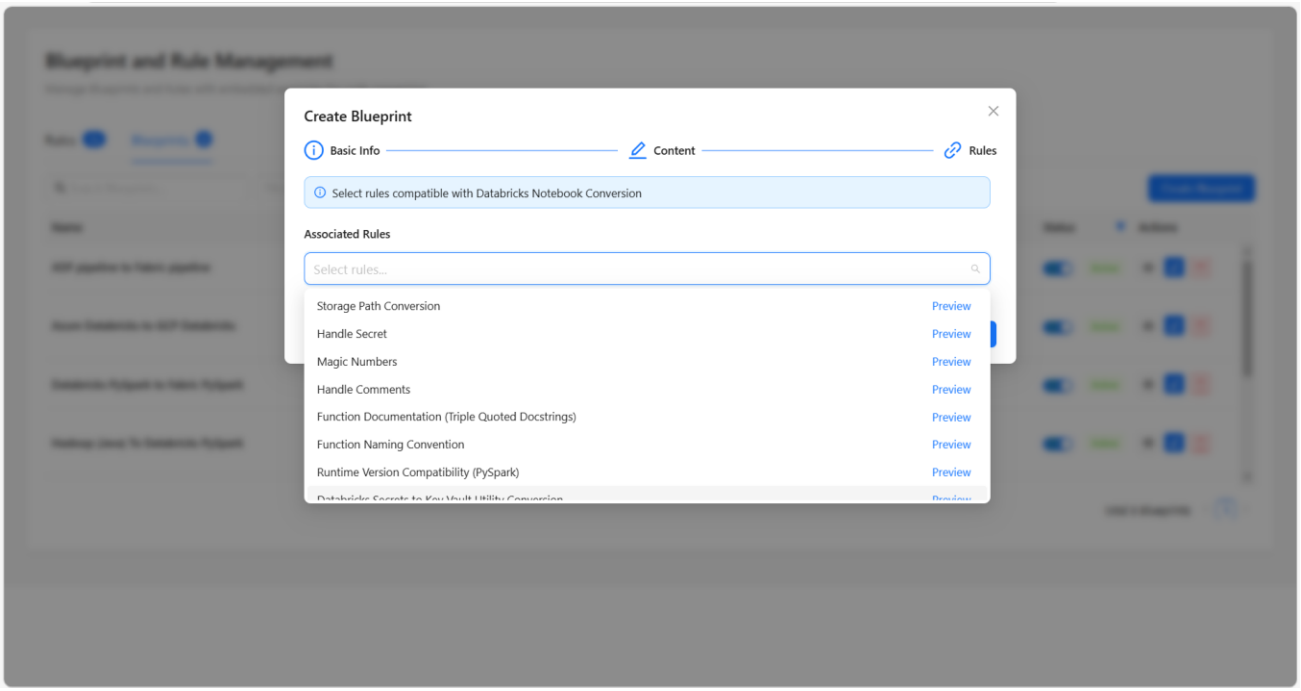

Image 16: Select associated rules

Associate Rules

User selects rules to attach with the Blueprint.

Rule Descriptions:

• **Databricks Widget to Parameter Conversion –** Converts Databricks widgets into parameterized inputs compatible with Fabric and standardized execution workflows.

• **Databricks Secrets to Key Vault Utility Conversion –** Replaces Databricks secret references with secure Key Vault-based access methods for improved credential management.

• **Databricks DBFS to Lakehouse Path Conversion –** Converts DBFS storage paths into Microsoft Fabric Lakehouse-compatible storage paths.

• **Databricks %run Magic to Python Imports –** Replaces Databricks notebook %run references with structured Python import statements for better modularity.

• **Databricks Preserve Function Calls and Signatures –** Preserves original function definitions, parameters, and invocation structures during conversion.

• **Databricks Display and Notebook Utility Conversion –** Converts Databricks-specific display utilities and notebook helper functions into Fabric-compatible implementations.

• **Databricks Shell and Package Install Comment-Out Pattern –** Detects and comments unsupported shell commands or package installation statements during migration.

• **Databricks Delta Lake Path and Operation –** Adjusts Delta Lake paths and operations to ensure compatibility with the target Fabric environment.

• **Function Naming Convention (Snake to Camel Case) –** Standardizes function and variable naming conventions for improved readability and coding consistency.

• **Function Documentation (Docstrings)** – Adds structured docstrings and documentation to functions for better maintainability and understanding.

• **Avoid Magic Numbers –** Replaces hardcoded numeric values with named constants or variables to improve readability and maintainability.

• **Code Indentation Standardization –** Applies consistent indentation and formatting standards to improve code readability and structure.

After selecting the rules, Click on **Create Blueprint**

Create New Rule

Click **"Create Rule"** and fill:

- **Name** – Rule name
- **Description** – What the rule does
- **Type** – Select rule type

Rule Types

- **Conversion** – Defines how specific code elements are transformed during migration.
- **Code Quality** – Enforces best practices like formatting, naming, and documentation.

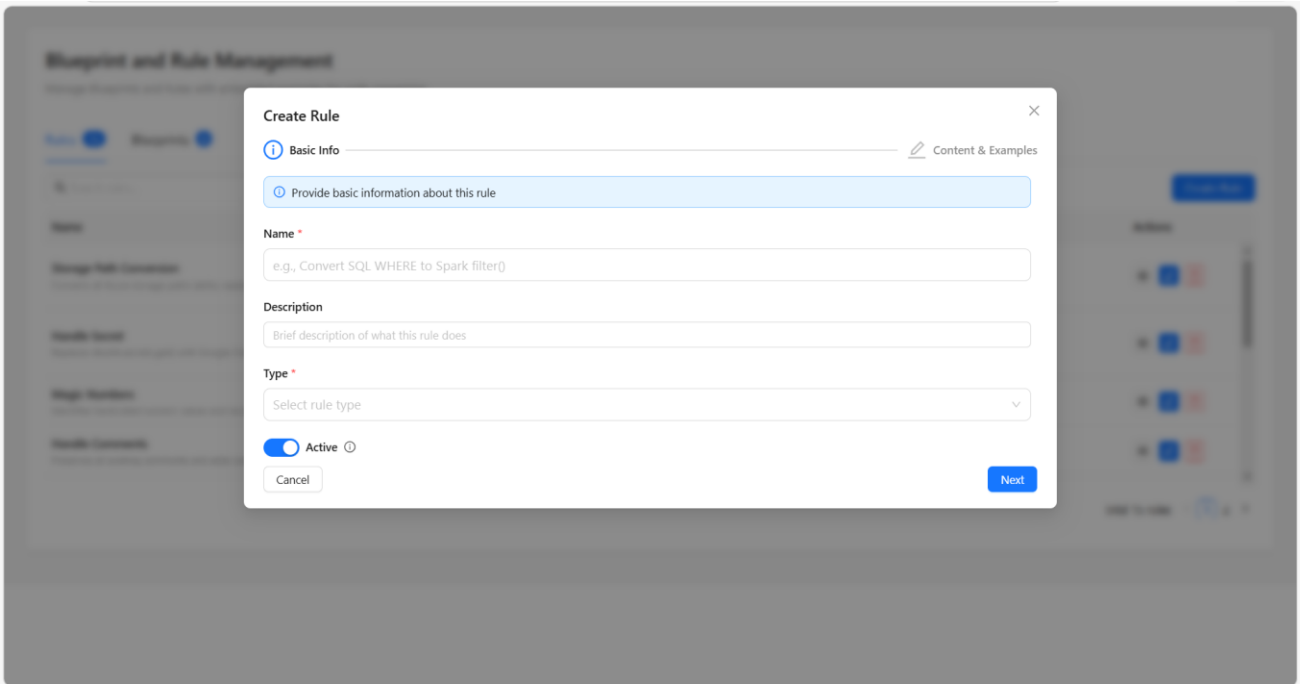

Image 17: Create Rule dialogue box

Rule Content & Examples

Rule Content

- Describes the transformation logic or coding standard.
- Can include conditions, instructions, or constraints.

Examples (Source → Target)

- Provide sample input and expected output.
- Helps the system understand exact transformation behavior.
- Improves accuracy and consistency.

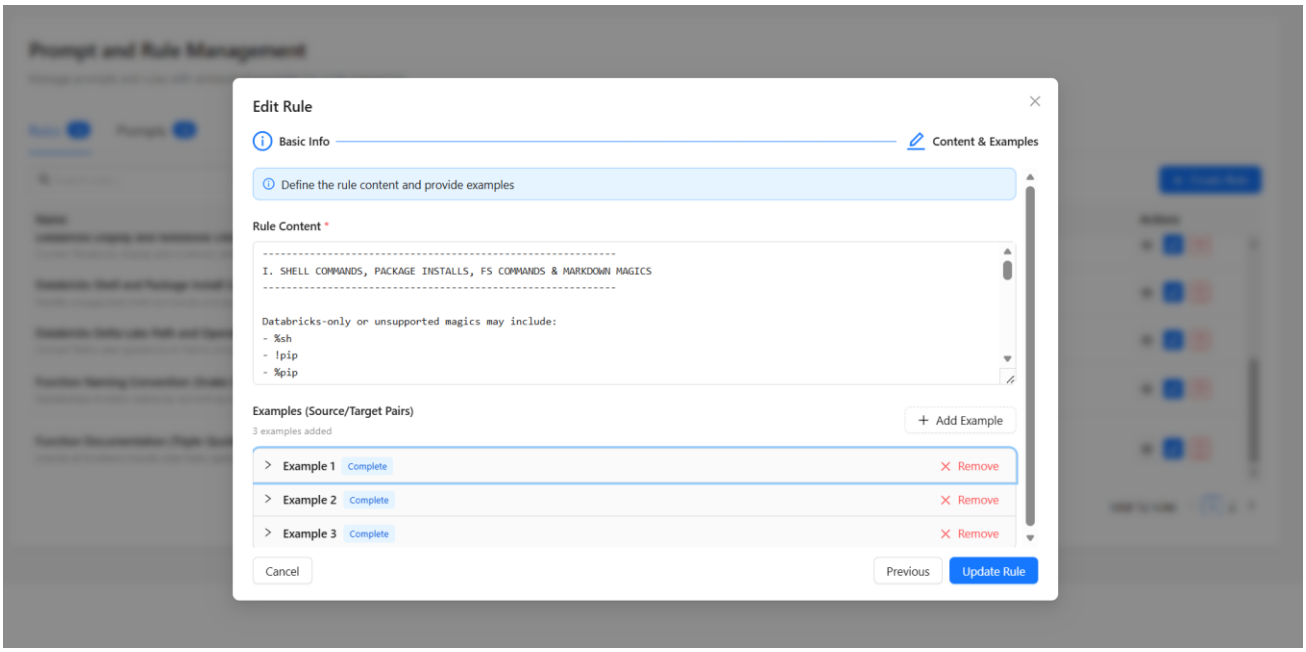

Image 18: Rule Content dialogue box

8. ADF Pipeline → Fabric Conversion (Step-by-Step Guide)

Create Workspace

- Click on **“Add Workspace”**
- Enter the following details:
  - **Name** → Pipeline
  - **Conversion Type** → ADF Pipeline → Fabric
  - **Blueprint** → Pipeline Conversion Blueprint
- Click **Submit**

The workspace will be created successfully.

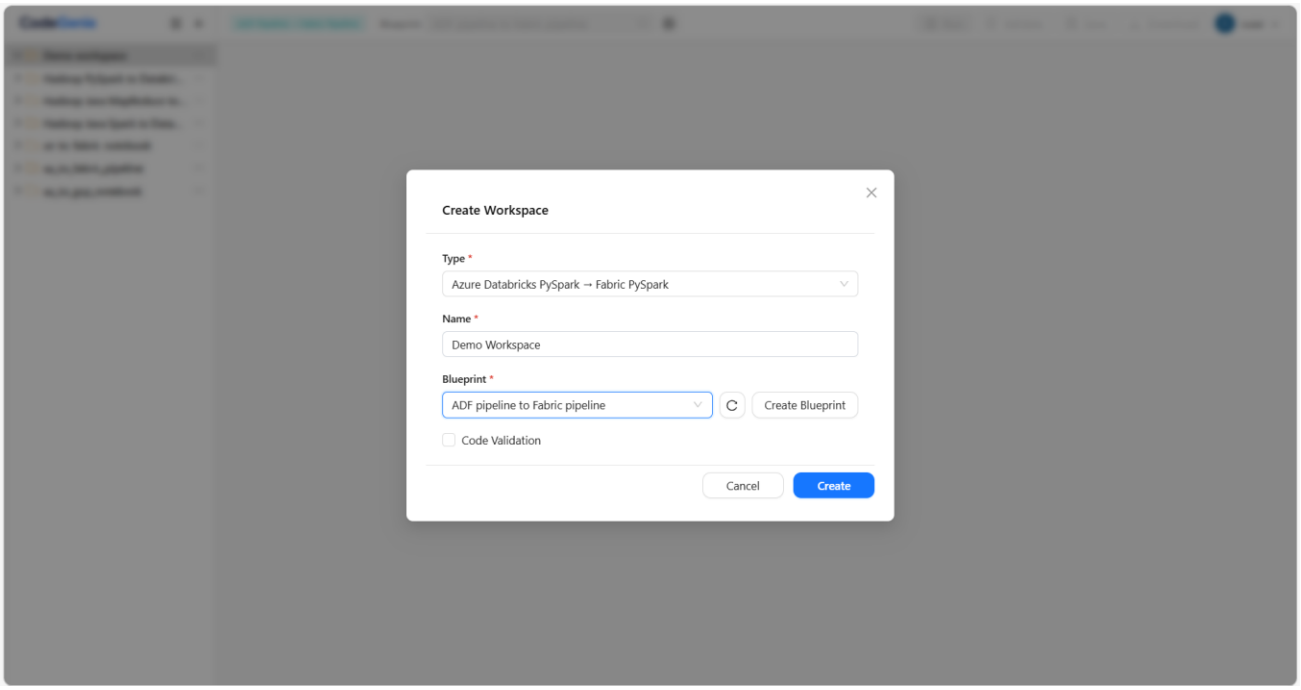

Image 19: ADF pipeline conversion workspace creation

Upload File

- Navigate inside the workspace
- Click **Open File / Add File**
- Upload the ADF pipeline JSON file

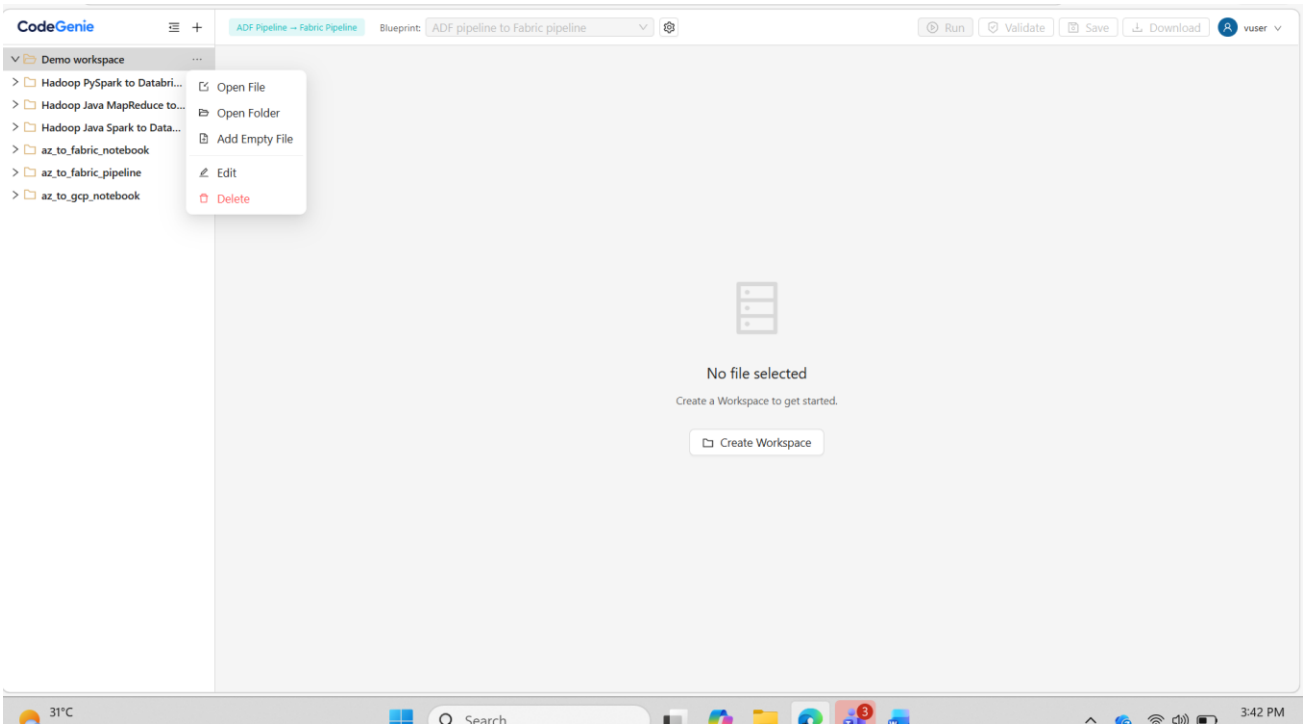

Image 20: Upload the ADF Pipeline JSON file

Open File

- Open the uploaded file in the editor
- You will see:
  - **Left panel** → Original ADF pipeline code
  - **Right panel** → Conversion output (initially empty)

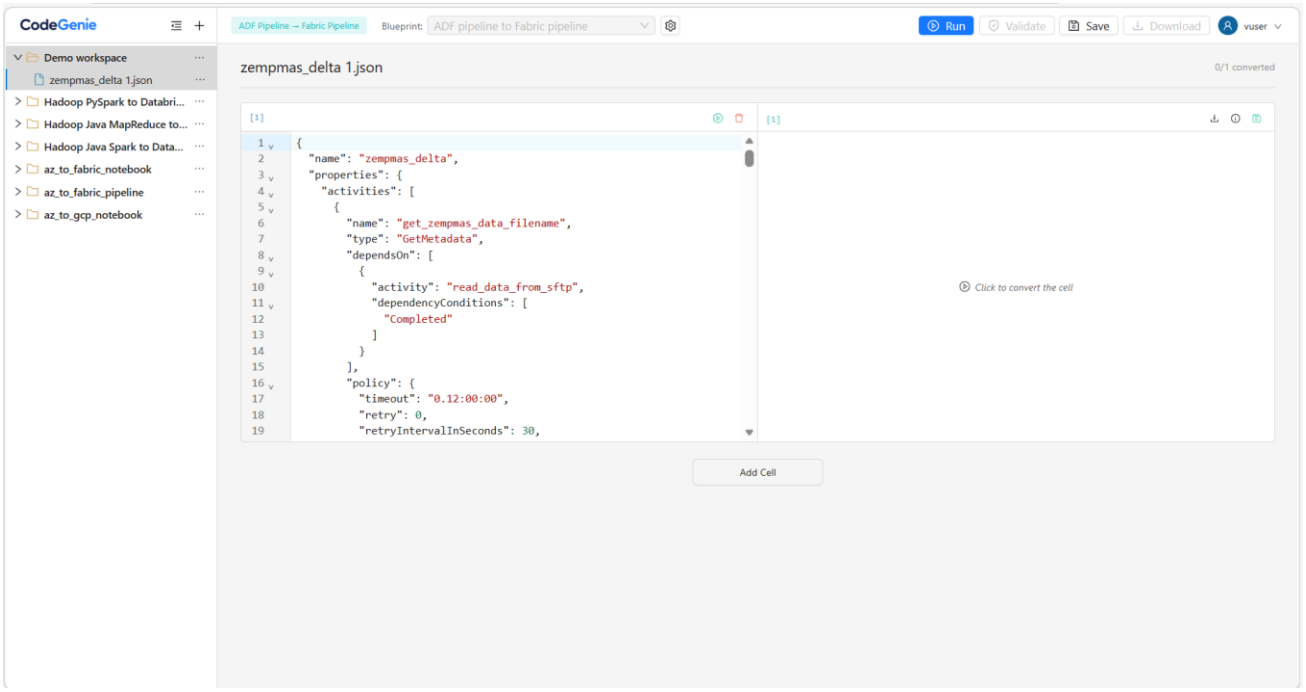

Image 21: Editor

Run Conversion

- Click the **Run** button
- The system converts the pipeline

After execution:

- The **Fabric-compatible pipeline code** is generated on the right side
- Status shows: Done

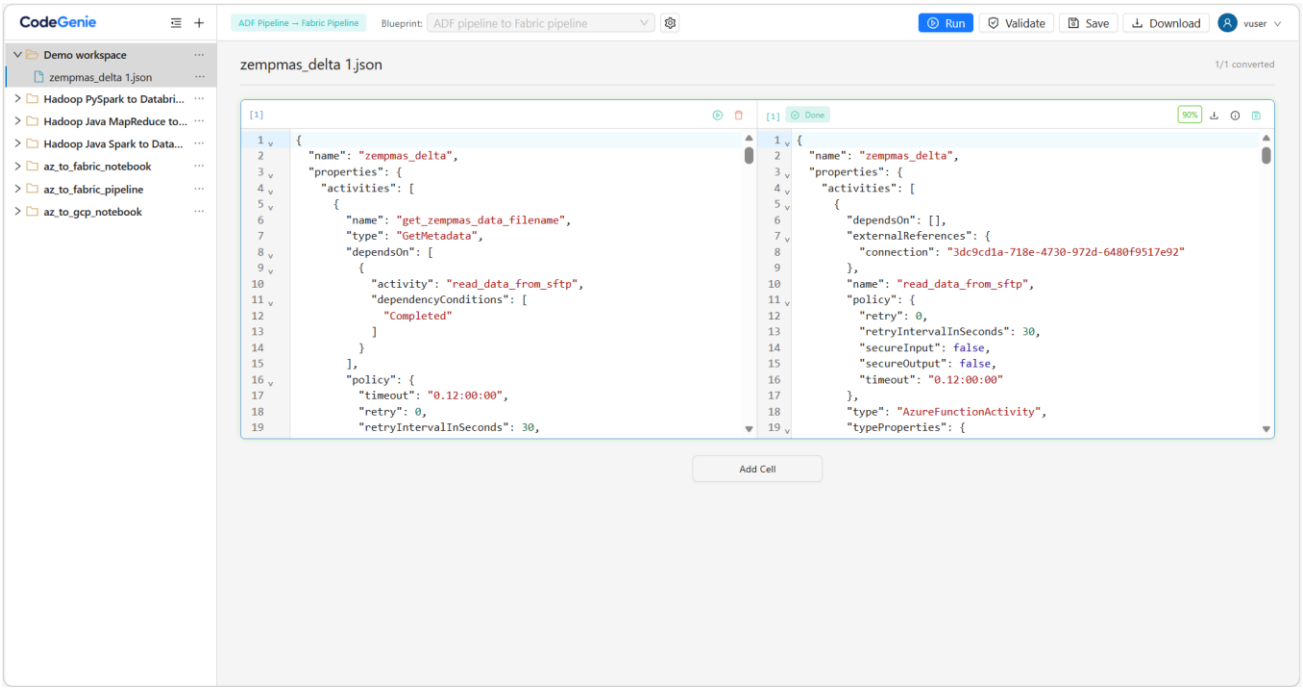

Image 22: Editor after successful run

View Output & Confidence

Users can view:

- Confidence Score (e.g., 90%) displayed for the converted output
- Converted code or pipeline output in the right panel
- Side-by-side comparison of source and converted content
- Conversion status and generated output for each cell

Available Actions

Users can perform the following actions:

- Run → Execute the conversion process
- Validate → Validate the generated output for errors or compatibility issues
- Save → Save the converted output
- Download → Download the generated output file
- Cell Info → View conversion details and related information for each cell

Check Cell Info

- Click **Cell Info**
- It displays: Pipeline converted successfully

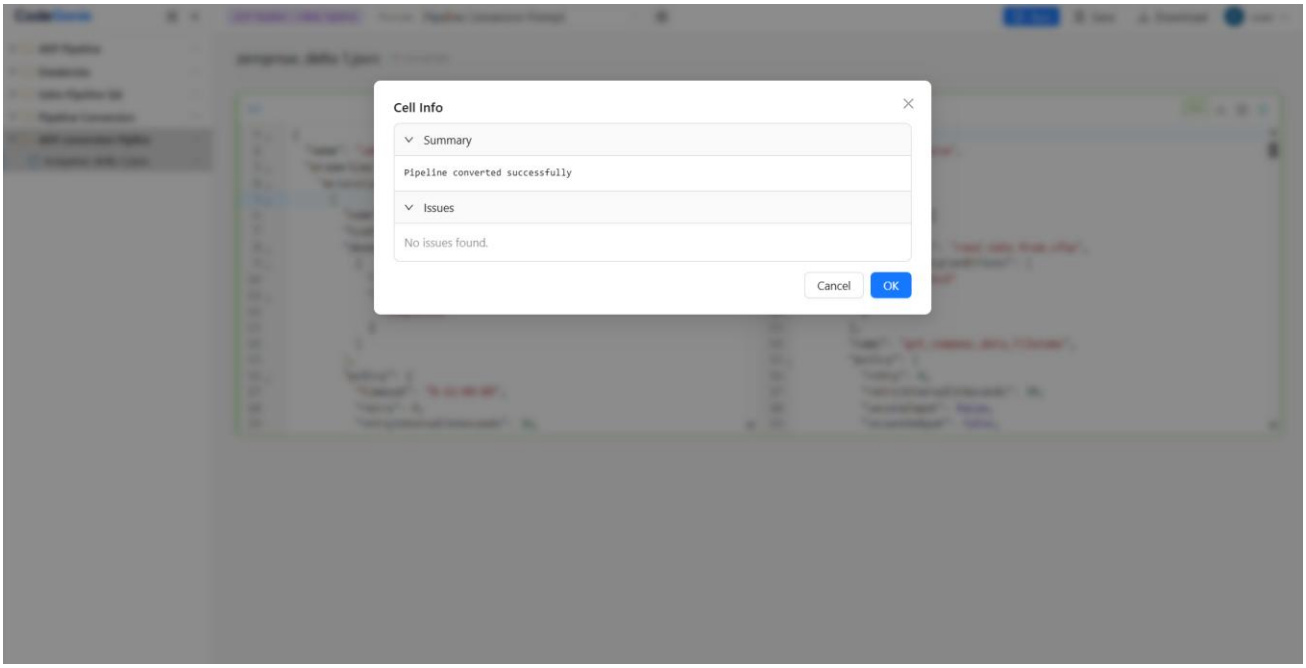

Image 23: Cell info dialogue box

Validate

- Click **Validate** button on top
- After validation, a status message is displayed such as:
  "Validation passed. No errors reported."
  " Validation issues or errors, if detected"

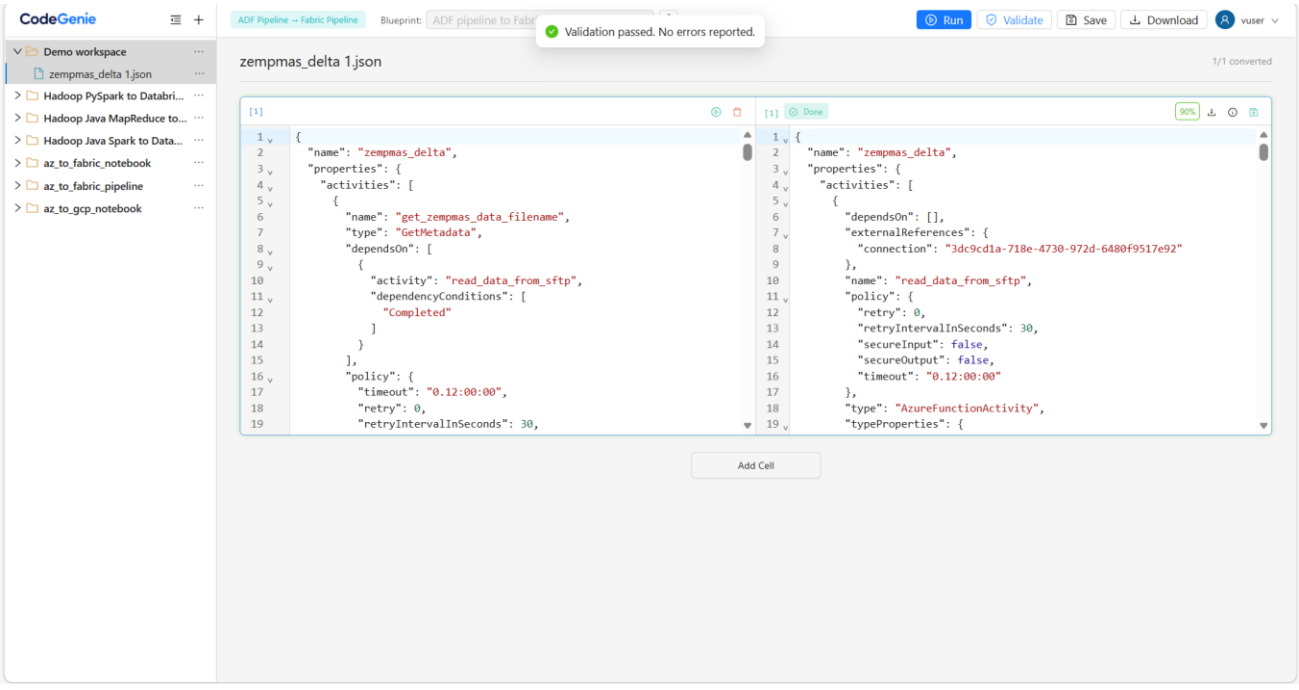

Image 24: Validation pop-up