Data ML Guide

Comprehensive guide for Data ML Process Flow Architecture

Data Intelligence Platform Team

Table of Contents

Welcome to the Data ML API Documentation

This site provides a comprehensive guide to the Data ML Platform API.

The platform is a machine learning service that allows users to ingest data, train predictive models, and perform inference. The core workflow involves defining data structures via templates, uploading and validating datasets, creating predictors, and running predictions.

Get Started

To view the complete user guide for all avilable features, please see the main documentation page:

- **View the Full User Guide**

To view the complete technical reference for all available endpoints, please see the main documentation page:

- **View the Full API Documentation**

Data ML - API Documentation

Version: 1.0.0
Date Created: September 17, 2025

Table of Contents

1. Introduction

1.1 Purpose of this Guide

This guide provides a comprehensive overview of the Data-ML Platform's Application Programming Interface (API). It is intended for developers and data scientists who need to interact with the platform programmatically. This document details the available API endpoints, their functionalities, request/response formats, and provides examples for seamless integration.

1.2 Overview of the Application

The Data-ML platform is a machine learning service that allows users to ingest data, train predictive models, and perform inference. The core workflow involves defining data structures via templates, uploading and validating datasets against these templates, and then creating and training predictors using Ray. Once a model is trained, the platform provides endpoints for both real-time (form-based) and bulk (batch) predictions, enabling a complete end-to-end machine learning lifecycle.

2. API Usage

2.1 Overview of the APIs

The following table provides a summary of the available API modules and their primary functions.

Module	API Endpoint	Description
Login	`POST /api/v1/login/access-token`	Handles user authentication to provide access tokens.
Users	`POST /api/v1/users/create-user`	Manages user creation and retrieval.
	`GET /api/v1/users/read-users`	
RBAC	`POST /api/v1/rbac/roles/`	Manages Role-Based Access Control by creating roles and permissions.
	`POST /api/v1/rbac/permissions/`	
Template	`POST /api/v1/template/`	Allows for the creation, retrieval, update, and deletion of data templates.
	`GET /api/v1/template/list`	
Dataset	`POST /api/v1/dataset/upload`	Handles dataset uploading, listing, validation, and statistical analysis.
	`GET /api/v1/dataset/list`	
File	`POST /api/v1/files/add`	Manages individual files within datasets.
	`GET /api/v1/files/list`	
Train	`POST /api/v1/train/`	Manages the creation and lifecycle of model training jobs.
	`GET /api/v1/train/list`	
Inference	`POST /api/v1/inference/`	Provides endpoints for running predictions using trained models.
	`POST /api/v1/inference/batch-prediction`	
Datalake	`GET /api/v1/datalake/schemas`	Provides utility endpoints to inspect the underlying datalake.
Utils	`GET /api/v1/utils/health-check/`	Provides utility endpoints for system health checks.

2.2 Authentication

Before making calls to most API endpoints, you must obtain a bearer token. The API expects an `Authorization` header with the value `Bearer <your_token>`. You can obtain this token by calling the `POST /api/v1/login/access-token` endpoint with valid credentials.

---

3 API Definitions

1. Module: Login APIs

1.1 Login Access Token

This API is used to authenticate a user via form data and receive an access token.

Endpoint: `/api/v1/login/access-token`
Method: `POST`

Request Body

Name	Description	Data Type	Omittable
`grant_type`	Grant Type	String	O
`username`	Username	String	M
`password`	Password	String	M
`scope`	Scope	String	O
`client_id`	Client ID	String	O
`client_secret`	Client Secret	String	O

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/login/access-token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'username=admin' \
--data-urlencode 'password=admin123'
```

Sample Response

```
{
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "token_type": "bearer"
}
```

1.2 Test Token

Validates the current user's token and returns their details.

Endpoint: `/api/v1/login/test-token`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/login/test-token' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "tenant_id": "a1b2c3d4-e5f6-7890-1234-567890abcdef",
    "first_name": "John",
    "last_name": "Doe",
    "email": "john.doe@example.com",
    "username": "johndoe",
    "id": 1,
    "is_active": true
}
```

1.3 Refresh Token

Refreshes an access token using a valid refresh token.

Endpoint: `/api/v1/refresh`
Method: `POST`

Query Parameters

Name	Description	Data Type	Omittable
refresh_token	The refresh token provided at login.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/refresh?refresh_token=eyJhbGciOiJIUzI1NiJ...'
```

Sample Response

```
{
    "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.new.token...",
    "token_type": "bearer"
}
```

1.4 Logout

Logs out a user by invalidating their refresh token.

Endpoint: `/api/v1/logout`
Method: `POST`

Query Parameters

Name	Description	Data Type	Omittable
refresh_token	The refresh token to invalidate.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/logout?refresh_token=eyJhbGciOiJIUzI1NiJ...'
```

Sample Response

```
{
    "status": 200,
    "message": "User successfully logged out"
}
```

1.5 Reset Password

Resets a user's password.

Endpoint: `/api/v1/reset-password`
Method: `POST`

Query Parameters

Name	Description	Data Type	Omittable
user_name	The username of the account.	String	M
new_password	The new password to set.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/reset-password?user_name=johndoe&new_password=NewSecurePassword123'
(200 OK)
```

Sample Response

```
{
    "status": 200,
    "message": "Password has been reset successfully"
}
```

2. Module: User APIs

2.1 Create User

This API creates a new user in the system.

Endpoint: `/api/v1/users/create-user`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
tenant_id	ID of the tenant the user belongs to.	String	M
first_name	First Name	String	O
last_name	Last Name	String	O
email	User's unique email address.	String (email)	M
username	User's unique username.	String	M
password	User's password.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/users/create-user' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
  "first_name": "jane",
  "last_name": "doe",
  "email": "jane.doe@example.com",
  "username": "janedoe",
  "password": "Password123!"
}'
```

2.2 Read Users

Retrieves a list of all users in the system.

Endpoint: `/api/v1/users/read-users/`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/users/read-users/' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "data": [
        {
            "id": 1,
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "first_name": "John",
            "last_name": "Doe",
            "email": "john.doe@example.com",
            "username": "johndoe",
            "is_active": true
        },
        {
            "id": 2,
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "first_name": "Jane",
            "last_name": "Doe",
            "email": "jane.doe@example.com",
            "username": "janedoe",
            "is_active": true
        }
    ],
    "count": 2
}
```

2.3 Retrieve User by ID

Retrieves the details for a single user by their unique ID.

Endpoint: `/api/v1/users/{user_id}`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
user_id	The unique ID of the user.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/users/1' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "id": 1,
    "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
    "first_name": "John",
    "last_name": "Doe",
    "email": "john.doe@example.com",
    "username": "johndoe",
```

```
    "is_active": true
}
```

3. Module: RBAC APIs

3.1 Create Role

This API creates a new role within a specific tenant.

Endpoint: `/api/v1/rbac/roles/`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
name	The name of the role (e.g., "data_scientist").	String	M
tenant_id	The unique ID of the tenant.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/rbac/roles/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "name": "data_scientist",
  "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c"
}'
```

Sample Response

```
{
    "id": 5,
    "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
    "name": "data_scientist"
}
```

3.2 Create Permission

This API creates a new permission within a specific tenant.

Endpoint: `/api/v1/rbac/permissions/`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
name	The name of the permission (e.g., "can_delete_dataset").	String	M
tenant_id	The unique ID of the tenant.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/rbac/permissions/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "name": "can_train_models",
  "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c"
}'
```

Sample Response

```
{
    "id": 25,
    "name": "can_train_models",
    "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c"
}
```

4. Module: Utils APIs

This section provides utility APIs for system monitoring and health checks.

4.1 Health Check

This API performs a simple health check of the service to confirm it is running and accessible.

Endpoint: `/api/v1/utils/health-check/`
Method: `GET`

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/utils/health-check/'
```

Sample Response

```
{
    "status": "ok",
    "message": "Service is healthy."
}
```

5. Module: Template APIs

This section details all APIs related to creating, retrieving, updating, and deleting data templates.

5.1 Create Template

This API creates a new data template with a defined schema.

Endpoint: `/api/v1/template/`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
`template_name`	A unique name for the template.	String	M
`template_schema`	Array of objects defining the columns.	Array	O
`template_schema.column_name`	The name of the column/header.	String	M
`template_schema.data_type`	The expected data type (e.g., String, Float).	String	M
`template_schema.default_value`	A default value for the column.	String	O

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/template/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "template_name": "Telecom Customer Usage",
  "template_schema": [
    {
      "column_name": "CustomerID",
      "data_type": "String",
      "default_value": "CUST-0000"
    },
    {
      "column_name": "DataUsageGB",
      "data_type": "Float",
      "default_value": "0.0"
    }
  ]
}'
```

Sample Response

```
{
    "status": 201,
    "message": "Template created successfully.",
    "id": 1
}
```

5.2 Get All Templates

This API retrieves a list of all available templates.

Endpoint: `/api/v1/template/list`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/template/list' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "total": 1,
    "data": [
        {
            "template_name": "Telecom Customer Usage",
            "id": 1,
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "no_of_columns": 2,
            "created_by": "admin",
            "created_at": "2025-09-18T10:00:00Z",
```

```
            "modified_at": "2025-09-18T10:00:00Z",
            "modified_by": "admin"
        }
    ]
}
```

5.3 Get Template

This API retrieves a single template and its full schema by its unique ID.

Endpoint: `/api/v1/template/{template_id}`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
template_id	The unique ID of the template.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/template/1' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "template_name": "Telecom Customer Usage",
    "id": 1,
    "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
    "no_of_columns": 2,
    "created_by": "admin",
    "created_at": "2025-09-18T10:00:00Z",
    "modified_at": "2025-09-18T10:00:00Z",
    "modified_by": "admin",
    "template_schema": [
        {
            "column_name": "CustomerID",
            "data_type": "String",
            "default_value": "CUST-0000"
        },
        {
            "column_name": "DataUsageGB",
            "data_type": "Float",
            "default_value": "0.0"
        }
    ]
}
```

5.4 Update Template

This API updates an existing template's name and schema.

Endpoint: `/api/v1/template/update/{template_id}`
Method: `PUT`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
template_id	The unique ID of the template to update.	Integer	M

Request Body (application/json)

Name	Description	Data Type	Omittable
template_name	The new name for the template.	String	M
template_schema	The new array of columns for the template.	Array	O

Sample Request

```
curl --location --request PUT 'http://localhost:8000/api/v1/template/update/1' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "template_name": "Telecom Customer Usage V2",
  "template_schema": [
    {
      "column_name": "CustomerID",
      "data_type": "String"
    },
    {
      "column_name": "DataUsageGB",
      "data_type": "Float"
    },
    {
      "column_name": "PlanType",
      "data_type": "String"
    }
  ]
}'
```

Sample Response

```
{
  "template_name": "Telecom Customer Usage V2",
  "template_schema": [
    {
      "column_name": "CustomerID",
      "data_type": "String",
      "default_value": null
    },
    {
      "column_name": "DataUsageGB",
      "data_type": "Float",
      "default_value": null
    },
    {
      "column_name": "PlanType",
      "data_type": "String",
      "default_value": null
    }
  ]
}
```

5.5 Delete Template

This API deletes a template by its unique ID.

Endpoint: `/api/v1/template/{template_id}`
Method: `DELETE`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
template_id	The unique ID of the template to delete.	Integer	M

Sample Request

```
curl --location --request DELETE 'http://localhost:8000/api/v1/template/1' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status_code": 200,
    "message": "Template deleted successfully"
}
```

6. Module: Dataset APIs

This section details all APIs related to uploading, managing, and analyzing datasets.

6.1 Upload Dataset

This API uploads data file(s) to create a new dataset and begin the validation process.

Endpoint: `/api/v1/dataset/upload`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`dataset_name`	A unique name for the new dataset.	String	M
`usage`	The intended usage (e.g., 'p' for prediction).	String	M
`template_id`	The ID of the template to validate against.	Integer	O
`iceberg_table_name`	The name of the Iceberg table if applicable.	String	O

Request Body (multipart/form-data)

Name	Description	Data Type	Omittable
`files`	The data file(s) to be uploaded.	File	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/dataset/upload?dataset_name=forecast-p11&template_id=2&usage=p' \
--header 'Authorization: Bearer <jwt_token>' \
--form 'files=@"/path/to/your/network_congestion_dataset.csv"'
```

Sample Response

```
{
    "dataset_id": 61,
    "dataset_name": "forecast-p11",
    "data_source": "csv",
    "files": [
        {
            "dataset_id": 61,
            "id": 80,
            "file_name": "network_congestion_dataset.csv",
            "created_at": "2025-09-18T18:30:00Z",
            "modified_at": "2025-09-18T18:30:00Z",
            "total_fields": 5,
            "total_records": 10000,
            "error_report": null
        }
    ]
}
```


6.2 List Datasets

This API retrieves a list of all available datasets.

Endpoint: `/api/v1/dataset/list`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/dataset/list' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "total": 1,
    "data": [
        {
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "dataset_name": "forecast-p11",
            "file_count": 1,
            "is_valid": true,
            "created_by": "admin",
            "template_id": 2,
            "data_source": "csv",
            "iceberg_table_name": "forecast_p11_iceberg",
            "id": 61,
            "template_name": "Network Congestion Template",
            "created_at": "2025-09-18T18:30:00Z",
            "modified_at": "2025-09-18T18:30:00Z",
            "dataset_usability": "85%",
            "usage": "p",
            "dataset_usage": "Prediction"
        }
    ]
}
```


6.3 List Validated Datasets

This API retrieves a list of datasets that have successfully passed the validation process.

Endpoint: `/api/v1/dataset/list_validated`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/dataset/list_validated' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
[
    {
        "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
        "dataset_name": "forecast-p11",
        "file_count": 1,
        "is_valid": true,
        "created_by": "admin",
        "template_id": 2,
        "id": 61
    }
]
```


6.4 Delete Dataset

This API deletes a dataset by its unique ID.

Endpoint: `/api/v1/dataset/delete/{dataset_id}`
Method: `DELETE`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
dataset_id	The unique ID of the dataset to delete.	Integer	M

Sample Request

```
curl --location --request DELETE 'http://localhost:8000/api/v1/dataset/delete/61' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "status_code": 200,
    "message": "Dataset deleted successfully"
}
```


6.5 Get Dataset by ID

This API retrieves detailed information for a single dataset by its unique ID.

Endpoint: `/api/v1/dataset/get_dataset_by_id`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
dataset_id	The unique ID of the dataset.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/dataset/get_dataset_by_id?dataset_id=61' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "dataset_name": "forecast-p11",
    "dataset_path": "/path/to/data/forecast-p11",
    "validated_path": "/path/to/validated/forecast-p11",
    "iceberg_table_name": "forecast_p11_iceberg",
    "data_source": "csv"
}
```

6.6 View Dataset

This API returns a preview of the data within a specified dataset.

Endpoint: `/api/v1/dataset/view`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
dataset_id	The unique ID of the dataset to view.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/dataset/view?dataset_id=61' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "preview": [
        { "timestamp": "2025-08-09T16:23:00Z", "tower_id": "TOWER_001", "traffic_load_gbps": 1.2, "connected_users": 150, "Congestion_Level_Percent": 15.5 },
        { "timestamp": "2025-08-09T16:24:00Z", "tower_id": "TOWER_001", "traffic_load_gbps": 1.3, "connected_users": 155, "Congestion_Level_Percent": 16.0 }
    ]
}
```

6.7 Generate Statistics

This API initiates a job to generate descriptive statistics for a dataset.

Endpoint: `/api/v1/dataset/generate_statistics`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
dataset_id	The unique ID of the dataset.	Integer	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/dataset/generate_statistics?dataset_id=61' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status": "success",
    "message": "Statistics generation job submitted successfully.",
    "job_id": "stat-job-12345"
}
```

7. Module: File APIs

This section details all APIs related to managing individual files within datasets.

7.1 Add Files to Dataset

This API adds one or more files to an existing dataset.

Endpoint: `/api/v1/files/add`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`dataset_id`	The unique ID of the dataset to add files to.	Integer	M

Request Body (multipart/form-data)

Name	Description	Data Type	Omittable
`files`	The data file(s) to be uploaded.	File	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/files/add?dataset_id=101' \
--header 'Authorization: Bearer <jwt_token>' \
--form 'files=@"/path/to/your/new_data.csv"'
```

Sample Response


```
{
    "dataset_id": 101,
    "dataset_name": "telecom_usage_q3",
    "data_source": "csv",
    "files": [
        {
            "dataset_id": 101,
            "id": 81,
            "file_name": "new_data.csv",
            "created_at": "2025-09-18T19:00:00Z",
            "modified_at": "2025-09-18T19:00:00Z",
            "total_fields": 15,
            "total_records": 5000,
            "error_report": null
        }
    ]
}
```


7.2 List Files in Dataset

Retrieves a list of all files associated with a specific dataset.

Endpoint: `/api/v1/files/list`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
dataset_id	The unique ID of the dataset to list files for.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/files/list?dataset_id=101' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "total": 2,
    "data": [
        {
            "dataset_id": 101,
            "id": 80,
            "file_name": "telecom_data.csv",
            "created_at": "2025-09-18T14:30:00Z",
            "modified_at": "2025-09-18T14:30:00Z",
            "total_fields": 15,
            "total_records": 10000,
            "error_report": null
        },
        {
            "dataset_id": 101,
            "id": 81,
            "file_name": "new_data.csv",
            "created_at": "2025-09-18T19:00:00Z",
            "modified_at": "2025-09-18T19:00:00Z",
            "total_fields": 15,
            "total_records": 5000,
            "error_report": null
        }
    ]
}
```


7.3 View File Content

Retrieves a preview of the content of a specific file.

Endpoint: `/api/v1/files/view`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
file_id	The unique ID of the file to view.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/files/view?file_id=80' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "file_id": 80,
    "content_preview": [
        "CustomerID,DataUsageGB,PlanType,Churn",
        "CUST-0001,10.5,Premium,False",
        "CUST-0002,2.1,Basic,True"
    ]
}
```

7.4 Delete File

Deletes a specific file by its unique ID.

Endpoint: `/api/v1/files/delete/{file_id}`
Method: `DELETE`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
file_id	The unique ID of the file to delete.	Integer	M

Sample Request

```
curl --location --request DELETE 'http://localhost:8000/api/v1/files/delete/81' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status_code": 200,
    "message": "File deleted successfully"
}
```

7.5 Download File

Downloads the raw content of a specific file.

Endpoint: `/api/v1/files/download/`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
file_path	The full storage path of the file.	String	M

Sample Request

```
# Note the -o flag to save the output to a local file
curl --location --request GET 'http://localhost:8000/api/v1/files/download/?file_path=/mnt/data/file_store/dataset_101/telecom_data.csv' \
--header 'Authorization: Bearer <jwt_token>' \
-o downloaded_telecom_data.csv
```

Sample Response

The raw content of the file is returned in the response body.

7.6 Get File Details by ID

Retrieves detailed metadata for a single file by its unique ID.

Endpoint: `/api/v1/files/get_file_by_id/`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
file_id	The unique ID of the file.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/files/get_file_by_id/?file_id=80' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "dataset_id": 101,
    "id": 80,
    "file_name": "telecom_data.csv",
    "created_at": "2025-09-18T14:30:00Z",
    "modified_at": "2025-09-18T14:30:00Z",
    "total_fields": 15,
    "total_records": 10000,
    "error_report": null,
    "csv_file_path": "/mnt/data/file_store/dataset_101/telecom_data.csv",
```

```
    "hdf_file_path": "/mnt/data/file_store/dataset_101/telecom_data.hdf"
}
```

8. Module: Train APIs

8.1 Create Training Job

This API creates a predictor and starts a new model training job using a validated dataset.

Endpoint: `/api/v1/train/` **Method:** `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
`dataset_id`	The ID of the validated dataset to train on.	Integer	M
`predictor_name`	A unique name for this predictor/model.	String	M
`domain`	The business domain (e.g., Churn, Forecast).	String	M
`problem_type`	The ML problem type (e.g., Classification).	String	M
`columns`	Array defining the role of each column.	Array	M
`is_incremental`	Flag for incremental training.	Boolean	O
`algorithm`	Specify the algorithm to be used.	String	O
`shap_enabled`	Flag to enable SHAP analysis.	Boolean	O

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/train/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "dataset_id": 101,
    "predictor_name": "churn_v1",
    "domain": "Churn",
    "problem_type": "Classification",
    "columns": [
        {
            "column_name": "CustomerID",
            "data_type": "String",
            "attribute_name": "Id"
        },
        {
            "column_name": "Churn",
            "data_type": "Boolean",
            "attribute_name": "Target"
        }
    ],
    "shap_enabled": true
}'
```

Sample Response

```
{
    "id": 5001,
    "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
    "predictor_name": "churn_v1",
    "problem_type": "Classification",
    "created_by": "admin",
    "dataset_id": 101,
    "created_at": "2025-09-18T14:30:00Z",
    "modified_at": "2025-09-18T14:30:00Z",
```

```
    "training_status": "submitted_to_ray"
}
```

8.2 Get Details for Training

Retrieves dataset schema and attributes required to configure a training job.

Endpoint: `/api/v1/train/get_details` **Method:** `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
`dataset_id`	The ID of the dataset.	Integer	M
`predictor_name`	The name for the new predictor.	String	M
`domain`	The business domain.	String	M
`problem_type`	The ML problem type.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/train/get_details' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "dataset_id": 101,
  "predictor_name": "churn_v1",
  "domain": "Churn",
  "problem_type": "Classification"
}'
```

Sample Response

```
{
    "dataset_id": 101,
    "predictor_name": "churn_v1",
    "domain": "Churn",
    "problem_type": "Classification",
    "dataset_schema": [
        { "column_name": "CustomerID", "data_type": "String" },
        { "column_name": "Tenure", "data_type": "Integer" },
        { "column_name": "Churn", "data_type": "Boolean" }
    ],
    "attributes": [
        "Id",
        "Target",
        "Feature"
    ]
}
```

8.3 List Training Jobs

Retrieves a list of all historical and active training jobs.

Endpoint: `/api/v1/train/list` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/list' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "total": 1,
    "data": [
        {
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "predictor_name": "churn_v1",
            "preprocess_id": 1,
            "problem_type": "Classification",
            "domain": "Churn",
            "created_by": "admin",
            "id": 5001,
            "dataset_id": 101,
            "dataset_name": "telecom_usage_q3",
            "data_source": "csv",
            "created_at": "2025-09-18T14:30:00Z",
            "modified_at": "2025-09-18T14:30:00Z",
            "algorithm": "XGBoost",
            "accuracy": 0.92,
            "ml_model_status": "completed",
            "train_status": "completed",
            "training_status": "Completed"
        }
    ]
}
```


8.4 Refresh Training Statuses

Refreshes the statuses of training jobs from the backend engine and returns the updated list.

Endpoint: `/api/v1/train/refresh` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/refresh' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "total": 1,
    "data": [
        {
            "tenant_id": "eb128f85-c955-4f2f-b109-0afa1aed409c",
            "predictor_name": "churn_v1",
            "preprocess_id": 1,
            "problem_type": "Classification",
            "domain": "Churn",
            "created_by": "admin",
            "id": 5001,
            "dataset_id": 101,
            "dataset_name": "telecom_usage_q3",
            "data_source": "csv",
            "created_at": "2025-09-26T10:30:00Z",
            "modified_at": "2025-09-26T10:45:00Z",
            "algorithm": "XGBoost",
            "accuracy": 0.92,
            "ml_model_status": "completed",
            "train_status": "completed",
            "training_status": "Completed"
        }
    ]
}
```

8.5 Delete Training Job

Deletes a training job and its associated model by its unique ID.

Endpoint: `/api/v1/train/delete/{id}` **Method:** `DELETE`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
`id`	The unique ID of the training job to delete.	Integer	M

Sample Request

```
curl --location --request DELETE 'http://localhost:8000/api/v1/train/delete/5001' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status_code": 200,
    "message": "Training job deleted successfully"
}
```

8.6 Update Model

Updates a model with a new preprocessing ID.

Endpoint: `/api/v1/train/update/` **Method:** `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
`train_id`	The ID of the training job.	Integer	M
`preprocess_id`	The ID of the new preprocessing step.	Integer	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/train/update/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "train_id": 19,
    "preprocess_id": 20
}'
```

Sample Response

```
{
    "data": "Model updated successfully",
    "status_code": 200,
```

```
    "message": "Success"
}
```

8.7 Training Result Callback

This is a callback endpoint for the Ray engine to post the results of a training job.

Endpoint: `/api/v1/train/result` **Method:** `POST`

Request Body (application/json)

Name	Description	Data Type	Omittable
`result`	The result object from the training job.	Object	M
`train_id`	The ID of the training job.	Integer	O
`preprocess_id`	The ID of the preprocessing job.	String	O

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/train/result' \
--header 'Content-Type: application/json' \
--data-raw '{
    "result": {
        "metrics": {
            "accuracy": 0.925
        },
        "checkpoint": "/path/to/model/checkpoint",
        "error": null,
        "path": "/path/to/model/output"
    },
    "train_id": 19,
    "preprocess_id": "preprocess-xyz"
}'
```

Sample Response

```
{
    "status": "success",
    "message": "Result received"
}
```

8.8 Download Training Schema

Downloads the input schema used for a specific training job.

Endpoint: `/api/v1/train/download-schema/{train_id}` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
`train_id`	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/download-schema/19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "schema": [
        { "column_name": "CustomerID", "data_type": "String" },
        { "column_name": "Tenure", "data_type": "Integer" },
        { "column_name": "Churn", "data_type": "Boolean" }
    ]
}
```

8.9 View Feature Importance

Retrieves the results of a previously generated feature importance analysis.

Endpoint: `/api/v1/train/view-feature-importance` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`train_id`	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/view-feature-importance?train_id=19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "feature_importance": {
        "Tenure": 0.45,
        "MonthlyCharges": 0.30,
        "ContractType": 0.15,
        "InternetService": 0.10
    },
    "status": "completed"
}
```

8.10 Generate Feature Importance

Starts a new job to calculate the feature importance (SHAP analysis) for a model.

Endpoint: `/api/v1/train/generate-feature-importance` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`train_id`	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/generate-feature-importance?train_id=19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status": "success",
    "message": "Feature importance job submitted successfully.",
    "shap_job_id": "shap-job-67890"
}
```

8.11 SHAP Result Callback

This is a callback endpoint for the Ray engine to post the results of a SHAP analysis.

Endpoint: `/api/v1/train/shap-result` **Method:** `POST`

Request Body (application/json)

Name	Description	Data Type	Omittable
`training_id`	The ID of the training job.	Integer	M
`result`	A JSON object containing the SHAP results.	Object	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/train/shap-result' \
--header 'Content-Type: application/json' \
--data-raw '{
    "training_id": 19,
    "result": {
        "Tenure": 0.45,
        "MonthlyCharges": 0.30,
        "ContractType": 0.15,
        "InternetService": 0.10
    }
}'
```

Sample Response

```
{
    "status": "success",
    "message": "SHAP result received"
}
```

8.12 Delete Feature Importance

Deletes the feature importance results for a specific training job.

Endpoint: `/api/v1/train/delete-feature-importance/{train_id}` **Method:** `DELETE`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Path Parameters

Name	Description	Data Type	Omittable
`train_id`	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request DELETE 'http://localhost:8000/api/v1/train/delete-feature-importance/19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status_code": 200,
    "message": "Feature importance results deleted successfully"
}
```

10.5 Get Model Summary

Retrieves the summary and performance metrics of a trained model.

Endpoint: `/api/v1/train/model-summary` **Method:** `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`train_id`	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/train/model-summary?train_id=5001' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "model_summary": {
        "model": "XGBoostClassifier",
        "accuracy": 0.925,
        "precision": 0.89,
        "recall": 0.94,
        "f1_score": 0.915
    },
    "confusion_matrix": [
        [950, 50],
        [30, 970]
    ]
}
```

9. Module: Inference APIs

This section details all APIs related to performing predictions using trained models. This includes real-time, batch, and forecast predictions.

9.1 Real-time Prediction

This API performs a real-time (single instance) prediction using a trained model.

Endpoint: `/api/v1/inference/`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
`train_id`	The ID of the trained model to use.	Integer	M
`preprocess_id`	The ID of the preprocessing step.	Integer	M
`features`	A JSON object containing the feature names and their values for prediction.	Object	M
`target`	The name of the target variable to predict.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/inference/' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
    "preprocess_id": 19,
    "train_id": 19,
    "features": {
        "Account_No": 1434355,
        "Base_Offer_Name": "B00010",
        "RMN_Counter": 12421314,
        "KDDI_Counter": 41235346,
        "Account_Activation_Date": "08-Aug-24",
        "Service_Barring": "true"
    },
    "target": "Churn"
}'
```

Sample Response

```
{
    "prediction": true,
    "accuracy": 0.925
}
```

9.2 Batch Prediction

This API starts a batch prediction job on an entire dataset using a trained model.

Endpoint: `/api/v1/inference/batchPrediction`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
train_id	The ID of the completed training job/model.	Integer	M
dataset_id	The ID of the dataset to run predictions on.	Integer	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/inference/batchPrediction' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "train_id": 19,
  "dataset_id": 21
}'
```

Sample Response

```
{
    "message": "Batch prediction job started successfully.",
    "status_code": 202
}
```

9.3 Get Form for Prediction

Retrieves the required schema/form fields for making a real-time prediction with a specific model.

Endpoint: `/api/v1/inference/form-based-prediction`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
id	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/inference/form-based-prediction?id=19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "id": 19,
    "headers": [
        {
            "column_name": "Account_No",
            "data_type": "Integer",
            "default_value": "0"
        },
        {
            "column_name": "Base_Offer_Name",
            "data_type": "String",
            "default_value": null
        }
    ],
    "target": "Churn",
    "preprocess_id": 19
}
```

9.4 Generate Forecast

This API runs a forecasting prediction based on a trained time-series model.

Endpoint: `/api/v1/inference/forecast`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Request Body (application/json)

Name	Description	Data Type	Omittable
train_id	The ID of the trained forecast model.	Integer	M
preprocess_id	The ID of the preprocessing step.	Integer	M
steps	The number of future steps to forecast.	Integer	M
frequency	The time frequency (e.g., 'Week', 'Day').	String	M
unique_id	The unique identifier for the time series.	String	M
dataset_id	The ID of the dataset to use for forecasting.	Integer	O

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/inference/forecast' \
--header 'Authorization: Bearer <jwt_token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "train_id": 2,
  "preprocess_id": 2,
  "steps": 4,
  "frequency": "Week",
  "unique_id": "CUST_0001"
}'
```

Sample Response

```
{
    "forecast": [
        { "date": "2025-10-05", "value": 150.5, "confidence": 0.95 },
        { "date": "2025-10-12", "value": 155.2, "confidence": 0.94 },
        { "date": "2025-10-19", "value": 153.8, "confidence": 0.93 },
        { "date": "2025-10-26", "value": 158.1, "confidence": 0.92 }
    ]
}
```

9.5 Get Batch Prediction Results

This API retrieves the results of a completed batch prediction job.

Endpoint: `/api/v1/inference/result`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
prediction_id	The unique ID of the prediction job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/inference/result?prediction_id=123' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status": "completed",
    "result_path": "/path/to/results/prediction_123.csv",
```

```
    "file_id": 81,
    "dataset_id": 21
}
```


9.6 List Predictions

This API retrieves a log of all predictions made using a specific trained model.

Endpoint: `/api/v1/inference/predictions`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
train_id	The unique ID of the training job.	Integer	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/inference/predictions?train_id=19' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response


```
{
    "data": [
        {
            "prediction_id": 123,
            "train_id": 19,
            "date": "2025-09-18T15:00:00Z",
            "dataset_id": 21,
            "predictor_name": "churn_v1",
            "prediction_type": "batch",
            "input_data": "dataset_id:21",
            "status": "completed",
            "no_of_records": 1000,
            "accuracy": 0.925,
            "problem_type": "Classification",
            "domain_name": "Churn"
        }
    ]
}
```


10. Module: Auth APIs

10.1 Validate Token

This API validates the provided bearer token to ensure it is active and not expired.

Endpoint: `/api/v1/validate-token`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/validate-token' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status": "success",
    "message": "Token is valid."
}
```

11. Module: Datalake APIs

11.1 List Schemas

This API retrieves a list of all available schemas (e.g., bronze, silver, gold) in the datalake.

Endpoint: `/api/v1/datalake/schemas`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/schemas' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
[
    "bronze",
    "silver",
    "gold",
    "default"
]
```

11.2 List Tables in Schema

Retrieves a list of all tables within a specific schema.

Endpoint: `/api/v1/datalake/schemas/tables`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
schema_name	The name of the schema to inspect.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/schemas/tables?schema_name=gold' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
[
    "customer_churn_predictions",
    "telecom_usage_forecasts",
```

```
    "user_profiles"
]
```

11.3 Get Table Columns

Retrieves the column names and data types for a specific table in a schema.

Endpoint: `/api/v1/datalake/tables/columns`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
schema_name	The name of the schema.	String	M
table_name	The name of the table to inspect.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/tables/columns?schema_name=gold&table_name=customer_churn_predictions' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
[
    { "column_name": "customer_id", "data_type": "varchar" },
    { "column_name": "prediction_date", "data_type": "timestamp" },
    { "column_name": "will_churn", "data_type": "boolean" },
    { "column_name": "churn_probability", "data_type": "double" }
]
```

11.4 Get Table Metadata

Retrieves detailed metadata for a specific table.

Endpoint: `/api/v1/datalake/table_metadata`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
schema_name	The name of the schema.	String	M
table_name	The name of the table.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/table_metadata?schema_name=gold&table_name=customer_churn_predictions' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "schema": "gold",
    "table": "customer_churn_predictions",
    "owner": "airflow",
    "created_time": "2025-09-10T10:00:00Z",
    "last_access_time": "2025-09-18T18:00:00Z",
    "location": "s3://datalake/gold/customer_churn_predictions",
    "format": "iceberg"
}
```

11.5 Get Numeric Stats

Calculates and retrieves basic statistics for a numeric column in a table.

Endpoint: `/api/v1/datalake/numeric_stats`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
schema_name	The name of the schema.	String	M
table_name	The name of the table.	String	M
column_name	The numeric column to analyze.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/numeric_stats?
schema_name=gold&table_name=customer_churn_predictions&column_name=churn_probability' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "column": "churn_probability",
    "count": 10000,
    "mean": 0.235,
    "std_dev": 0.15,
    "min": 0.01,
    "max": 0.99,
    "median": 0.21
}
```

11.6 Get Distinct Count

Calculates and retrieves the count of distinct values in a column.

Endpoint: `/api/v1/datalake/distinct-count`
Method: `GET`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
schema	The name of the schema.	String	M
table	The name of the table.	String	M
column	The column to analyze.	String	M

Sample Request

```
curl --location --request GET 'http://localhost:8000/api/v1/datalake/distinct-count?schema=gold&table=user_profiles&column=country' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "column": "country",
    "distinct_count": 85
}
```

12. Module: Ray APIs

12.1 Terminate Ray Job

This API is used to manually terminate a running Ray job, such as a training or statistics generation task.

Endpoint: `/api/v1/ray/terminate_ray_job`
Method: `POST`

Request Headers

Name	Description	Data Type	Omittable
Authorization	The bearer token.	String	M

Query Parameters

Name	Description	Data Type	Omittable
`ray_job_id`	The unique ID of the Ray job to terminate.	String	M
`usage`	The context/usage of the Ray job.	String	M

Sample Request

```
curl --location --request POST 'http://localhost:8000/api/v1/ray/terminate_ray_job?ray_job_id=ray-job-abc123&usage=training' \
--header 'Authorization: Bearer <jwt_token>'
```

Sample Response

```
{
    "status": "success",
    "message": "Termination request for Ray job 'ray-job-abc123' has been sent."
}
```

4. API Status Codes

This section lists common HTTP status codes returned by the API endpoints and their meanings.

Status Code	Meaning
200	OK - Successful request
201	Created - Resource successfully created
202	Accepted - Request accepted for processing
400	Bad Request - Invalid input or parameters
401	Unauthorized - Authentication required or failed
403	Forbidden - Insufficient permissions
404	Not Found - Resource not found
409	Conflict - Resource conflict
422	Unprocessable Entity - Validation error
500	Internal Server Error - Unexpected error

5. Version

Current Version: 1.0.0
Last Updated: 22 September 2025

Machine Learning Platform Documentation

Version: 1.0.0
Date Created: January 02, 2026

Table of Contents

1. Purpose of the Platform

1.1 Overview

This platform exists to provide a single, consistent way to build and operate machine learning models across the organization.

Instead of individual teams handling data preparation, model training, and prediction in their own scripts or services, the platform centralizes these workflows behind well-defined APIs and controlled execution paths. This allows teams to focus on model logic and use-case design, while the platform takes responsibility for data validation, preprocessing consistency, execution, and tracking.

From a technical perspective, the system acts as an orchestration layer over data, preprocessing pipelines, model training jobs, and inference workloads. From a user's perspective, it offers a structured, repeatable process for moving from raw datasets to production-ready predictions.

1.2 Why the Platform Exists

As machine learning usage scales, a few recurring problems tend to emerge: inconsistent preprocessing between training and inference, difficulty reproducing past results, long-running jobs blocking application services, and limited visibility into what data or configuration produced a given model.

This platform is designed to address those problems directly.

All training and prediction operations run through a controlled lifecycle. Datasets are explicitly marked for training or prediction use. Feature mappings, target selection, and preprocessing steps are recorded at training time and reused verbatim during inference. Model artifacts and metrics are stored alongside the metadata needed to understand how they were produced.

The result is a system where models can be trained, evaluated, reused, and updated without relying on undocumented assumptions or external scripts.

1.3 What the Platform Manages

The platform manages the full machine learning workflow end-to-end.

It starts with data definition. Before any model can be trained, the structure of the data is defined using templates. These templates describe column names, data types, and default values, and they serve as the contract between datasets, preprocessing logic, and models. This approach ensures that training and prediction data always conform to an expected schema.

Once data is defined and validated, the platform manages model training. Training requests capture not only the algorithm and problem type, but also how each column is interpreted (for example, which column is the target, which represents time, or which acts as a unique identifier). The platform then coordinates preprocessing, model execution, metric collection, and artifact storage.

Finally, the platform manages prediction and forecasting. Predictions are always executed using the preprocessing state generated during training, ensuring consistency between training and inference. Both real-time (form-based) and batch prediction workflows are supported, as well as time-series forecasting with optional exogenous variables.

1.4 Design Approach

The platform is intentionally conservative in how it handles machine learning workflows.

Validation is enforced early and often. Invalid configurations are rejected before any compute-intensive work begins. This includes checks on dataset usage, problem type compatibility, feature mappings, and algorithm constraints. While this may feel restrictive, it significantly reduces runtime failures and ambiguous behavior.

Long-running operations such as training, batch prediction, and SHAP computation are executed asynchronously. The API layer remains responsive, while execution is handled by distributed workers. Results are reported back through well-defined callbacks and persisted in the database.

Throughout the system, emphasis is placed on traceability. Every dataset, training run, and prediction can be traced back to the inputs and configuration that produced it. This is essential for debugging, auditing, and long-term model maintenance.

1.5 Intended Usage

The platform is intended for teams that need to train and operate machine learning models in a controlled, repeatable manner.

It supports iterative model development, where models are retrained or incrementally updated as new data becomes available. It also supports large-scale batch inference workflows and forecasting use cases that require careful handling of time and frequency.

By enforcing structure and consistency, the platform allows machine learning workflows to scale without becoming brittle or opaque.

2. High-Level Architecture and Core Components