2025

EarthScape - Project Report

**A tree in a bubble

AI-generated content may be incorrect.**

Table of Contents

[Student Information 3](#_Toc208242533)

[Management 3](#_Toc208242534)

[Specification 4](#_Toc208242535)

[Tasks Completed by Each Student 10](#_Toc208242536)

[Documentation Completed by Each Student 11](#_Toc208242537)

[**1. Project Report** 13](#_Toc208242538)

[**1.1 Authentication and Authorization** 13](#_Toc208242539)

[**Registering as a User (Analyst)** 13](#_Toc208242540)

[**Logging In** 13](#_Toc208242541)

[**Logging Out** 15](#_Toc208242542)

[**1.2 Data Ingestion** 15](#_Toc208242543)

[**Purpose** 15](#_Toc208242544)

[**Dataset Management** 20](#_Toc208242545)

[**1.3 Data Storage** 22](#_Toc208242546)

[**Implementation** 22](#_Toc208242547)

[**Optimization Strategies** 24](#_Toc208242548)

[**1.4 Data Processing** 26](#_Toc208242549)

[**1.5 Real-time Data Processing** 30](#_Toc208242550)

[**Steps** 30](#_Toc208242551)

[**Access Control** 33](#_Toc208242552)

[**Error Handling** 33](#_Toc208242553)

[**1.6 Anomaly Training** 33](#_Toc208242554)

[**Steps** 33](#_Toc208242555)

[**1.7 Forecast Training** 35](#_Toc208242556)

[**1.8 Correlation Analysis** 36](#_Toc208242557)

[**Features** 36](#_Toc208242558)

[**1.10 Weather Details** 39](#_Toc208242559)

[**Features** 39](#_Toc208242560)

[**1.11 Weather ChatBot** 41](#_Toc208242561)

[**Features** 41](#_Toc208242562)

[**1.12 Admin Management** 43](#_Toc208242563)

[**Features** 43](#_Toc208242564)

[**1.13 Feedback Management** 46](#_Toc208242565)

[**Features** 46](#_Toc208242566)

# Student Information

|  |  |
| --- | --- |
| Student ID | Student Name |
| Student1420981 | **Muhammad Saad Karim** |
| Student1421220 | **Tayyaba** |
| Student1368985 | **Mahnoor** |
| Student1418404 | **Laraib** |
| Student1335767 | **Lamia Raza** |
| Student1418410 | **Albina** |

# Management

|  |  |
| --- | --- |
| Role | Student Name |
| Faculty | **Mam Wajiha** |
| Coordinator | **Mam Sana** |

# Specification

**Introduction**

**The thirst for learning, upgrading technical skills and applying the concepts in real life environment at a fast pace is what the industry demands from IT professionals today. However busy work schedules, far-flung locations, unavailability of convenient time-slots pose as major barriers when it comes to applying the concepts into realism. And hence the need to look out for alternative means of implementation in the form of laddered approach.**

**The above truly pose as constraints especially for our students too! With their busy schedules, it is indeed difficult for our students to keep up with the genuine and constant need for integrated application which can be seen live especially so in the field of IT education where technology can change on the spur of a moment. Well, technology does come to our rescue at such times!!**

**Keeping the above in mind and in tune with our constant endeavour to use Technology in our training model, we at Aptech have thought of revolutionizing the way our students learn and implement the concepts using tools themselves by providing a live and synchronous eProject learning environment!**

**So what is this eProject?**

**eProject is a step by step learning environment that closely simulates the classroom and Lab based learning environment into actual implementation. It is a project implementation at your fingertips!! An electronic, live juncture on the machine that allows you to**

**• Practice step by step i.e. laddered approach.**

**• Build a larger more robust application.**

**• Usage of certain utilities in applications designed by user.**

**• Single program to unified code leading to a complete application.**

**• Learn implementation of concepts in a phased manner.**

**• Enhance skills and add value.**

**• Work on real life projects.**

**• Give a real life scenario and help to create applications more complicated and useful.**

**• Mentoring through email support.**

**The students at the centre are expected to complete this eProject and send complete documentation with source code to eProjects Team**

**Looking forward to a positive response from your end!!**

**Objectives of the project**

**The Objective of this program is to give a sample project to work on real life projects. These applications help you build a larger more robust application.**

**The objective is not to teach you the concepts but to provide you with a real life scenario and help you create applications using the tools.**

**You can revise them before you start with the project.**

**It is very essential that a student has a clear understanding of the subject.**

**Kindly get back to eProjects Team in case of any doubts regarding the application or its objectives.**

**Background**

**Climate change is a global challenge that demands comprehensive analysis and proactive measures to mitigate its impacts. The increase in greenhouse gas emissions, deforestation, and changes in land use have led to unprecedented shifts in climate patterns, resulting in more frequent and severe weather events. To address this complex issue, the integration of big data technologies, particularly Hadoop, becomes crucial for processing, analyzing, and interpreting the vast amount of climate-related data available.**

**The project is initiated for a EarthScape Climate Agency, dedicated to monitoring and addressing climate change issues. The agency is responsible for collecting vast amounts of climate-related data from various sources, including satellites, weather stations, and environmental sensors. They seek a robust solution to process, analyze, and visualize this data to gain actionable insights for informed decision-making.**

**Functional Requirements**

**User Authentication and Authorization:**

**Implement a secure authentication system for users with different roles (e.g., administrators, analysts).**

**Define access controls to restrict data access based on user roles and responsibilities.**

**Data Ingestion:**

**The system should support the ingestion of diverse climate-related datasets, including satellite imagery, weather station records, and environmental sensor data.**

**Implement mechanisms to handle both historical and real-time data sources.**

**Ensure compatibility with common data formats used in climate science.**

**Data Storage:**

**Utilize the Hadoop Distributed File System (HDFS) for scalable and fault-tolerant storage of large climate datasets.**

**Implement data partitioning and organization strategies to optimize retrieval and processing.**

**Data Processing:**

**Implement Hadoop MapReduce jobs for parallel processing of climate data across distributed nodes.**

**Develop algorithms for the identification of climate patterns, anomalies, and correlations.**

**Include mechanisms to handle missing or incomplete data gracefully.**

**Real-time Data Processing:**

**Integrate real-time data streaming capabilities.**

**Ensure seamless integration with batch processing for a comprehensive analysis.**

**Machine Learning Models:**

**Develop machine learning models for predictive analysis of climate trends and impacts.**

**Include algorithms for anomaly detection, trend prediction, and correlation analysis.**

**Regularly update and refine models based on the latest available data.**

**Data Visualization:**

**Create interactive dashboards.**

**Develop visual representations of climate patterns, anomalies, and predictions.**

**Provide customizable and user-friendly interfaces for stakeholders to explore data**

**Notifications and Alerts:**

**Set up automated notifications and alerts for stakeholders based on predefined thresholds for climate anomalies or significant events.**

**Enable configurable alerting mechanisms to notify users in real-time.**

**Feedback and Support:**

**A support system for users to contact for assistance, report issues, and provide feedback.**

**Non-Functional Requirements**

**Performance:**

**Implement monitoring tools to track system performance, resource utilization, and data processing times.**

**Include optimization strategies for enhancing the overall efficiency of the system.**

**Data Security:**

**Implement encryption mechanisms to secure sensitive climate data during storage and transmission.**

**Ensure compliance with relevant data protection regulations and standards.**

**Reliability:**

**Uptime: The application should aim for a minimum of 99% uptime, with scheduled maintenance communicated in advance.**

**Data Backup: Regular automated data backups must be performed to prevent data loss in case of system failures.**

**Scalability:**

**The architecture should support horizontal scaling to accommodate increased data volumes and processing demands.**

**Implement load balancing mechanisms for optimal resource utilization.**

**Performance Monitoring:**

**Implement monitoring tools to track system performance, resource utilization, and data processing times.**

**Include optimization strategies for enhancing the overall efficiency of the system.**

**Compliance and Standards:**

**Ensure adherence to relevant environmental data standards and protocols.**

**Comply with industry best practices for big data processing and analytics.**

**Documentation:**

**User Documentation: Provide user guides, FAQs, and tutorials to help users understand and navigate the application.**

**Developer Documentation: Provide thorough documentation for system architecture, data processing workflows, and machine learning models. Maintain developer documentation to assist in further development and maintenance.**

**Video: Provide video displaying complete working of the application.**

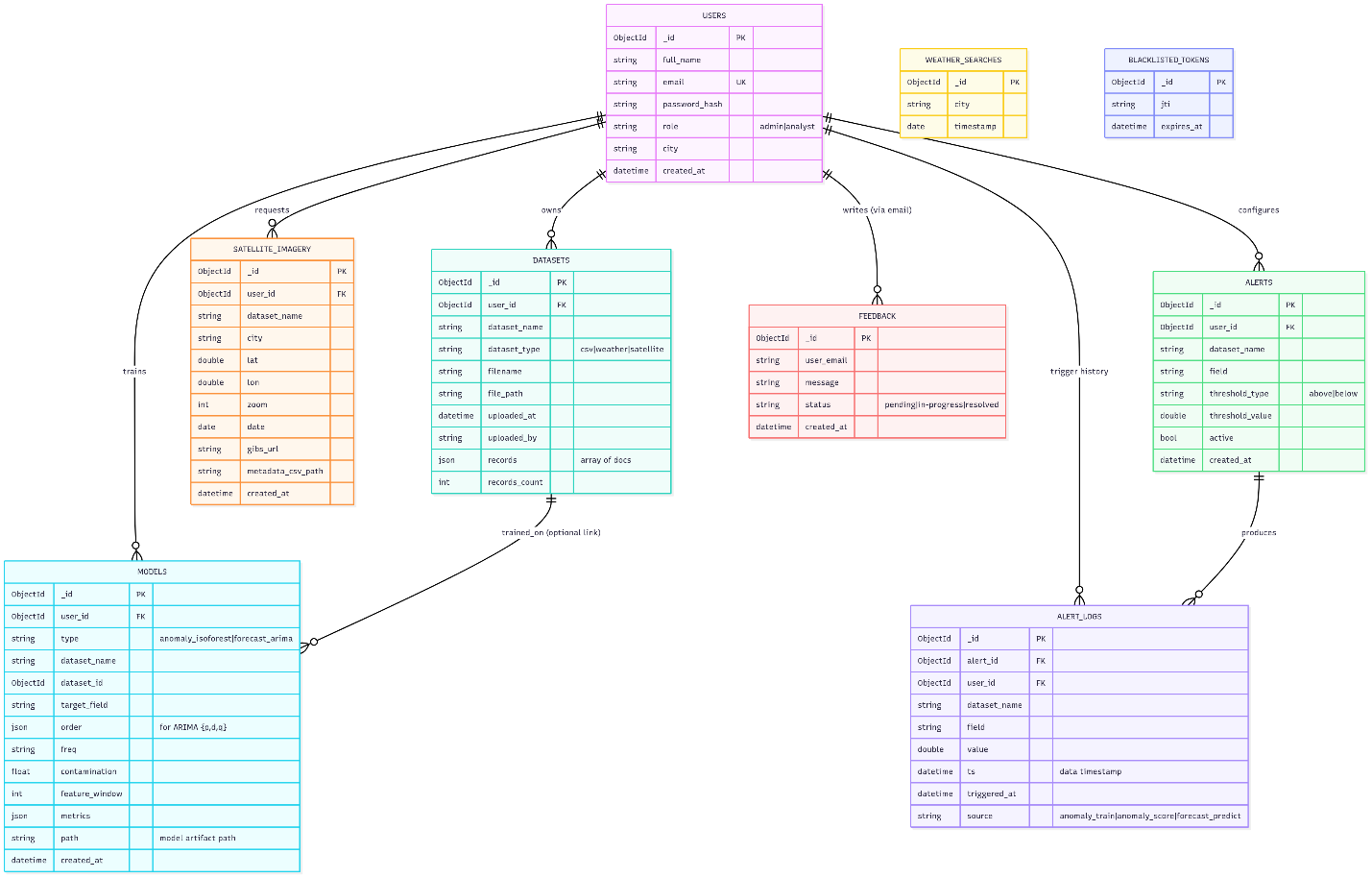
# Tasks Completed by Each Student

|  |  |
| --- | --- |
| Module / Task | Completed By |
| Authentication & Authorization | **Tayyaba (1421220)** |
| Data Ingestion (CSV, Weather, Satellite) | **Saad (1420981)** |
| Data Storage (MongoDB Atlas, collections, optimization) | **Saad (1420981)** |
| Data Processing (count, average, group by, pattern) | **Tayyaba (1421220)** |
| Real-time Data Processing (live feeds, charts) | **Saad (1420981)** |
| Anomaly Training (Isolation Forest model) | **Tayyaba (1421220)** |
| Forecast Training (ARIMA model) | **Saad (1420981)** |
| Correlation Analysis (ACF, Cross, Matrix) | **Tayyaba (1421220)** |
| Weather Details (API integration & dashboard) | **Saad (1420981)** |
| Weather ChatBot (frontend + API) | **Tayyaba (1421220)** |
| Admin Management (add/update/delete admins) | **Saad (1420981)** |
| Feedback Management (submit/view/update feedback) | **Tayyaba (1421220)** |
| Alert And Notification | **Saad (1420981)** |

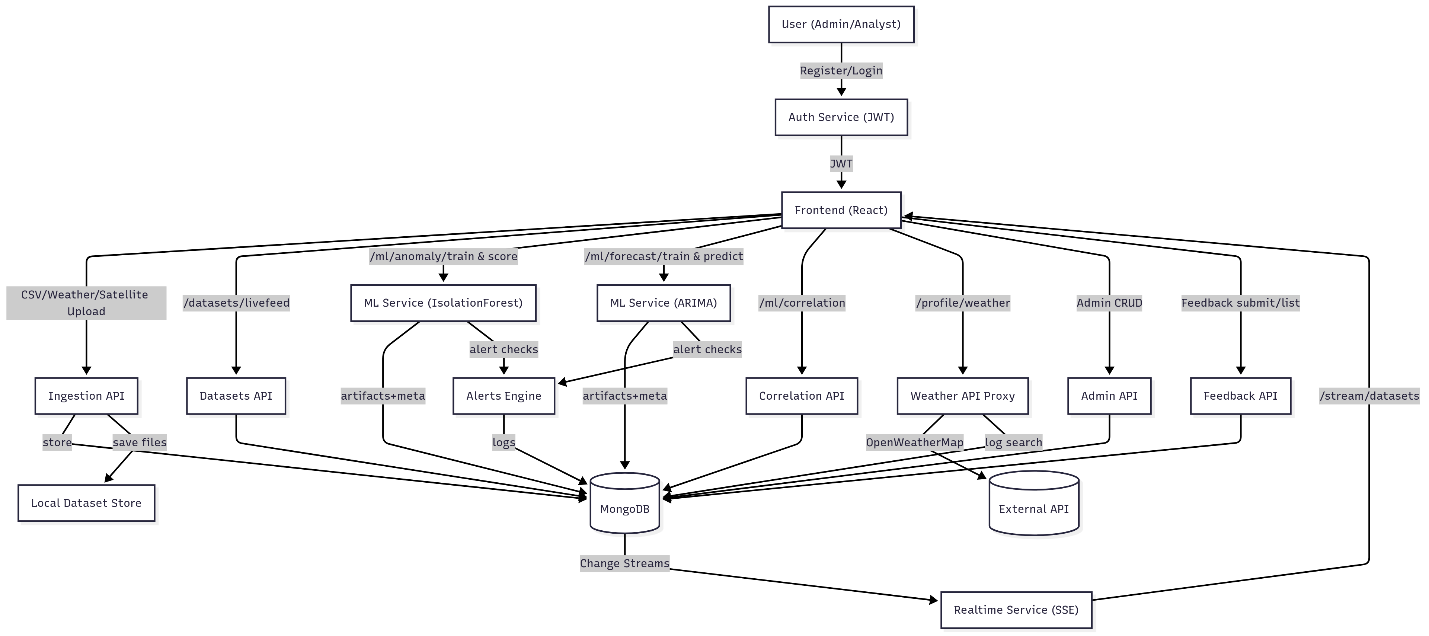
# Documentation Completed by Each Student

|  |  |
| --- | --- |
| Documentation Type | Completed By |
| User Guide | **Tayyaba (1421220)** |
| Project Report | **Saad (1420981)** |
| Developer Guide | **Tayyaba (1421220)** |
| Presentation | **Saad (1420981)** |

**ERD**

****

**System Flow**

****

**Real-Time SSE Sequence**

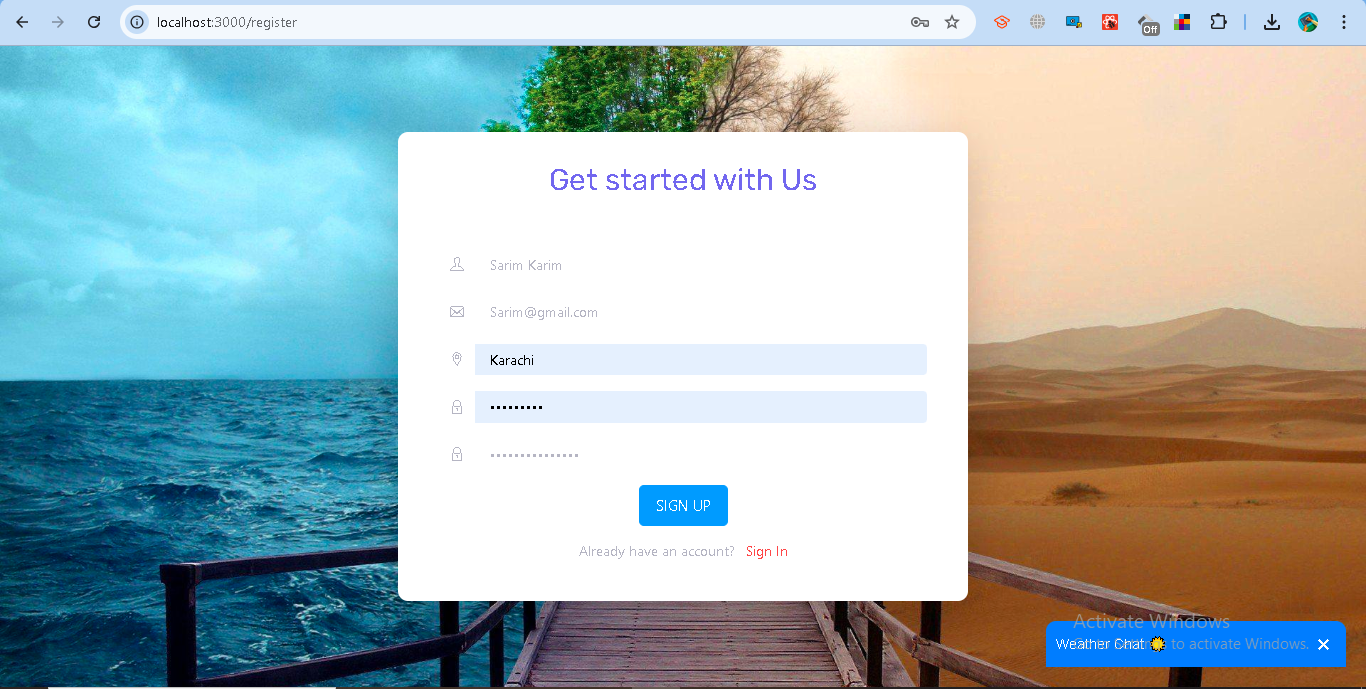
**A diagram of a computer

AI-generated content may be incorrect.**

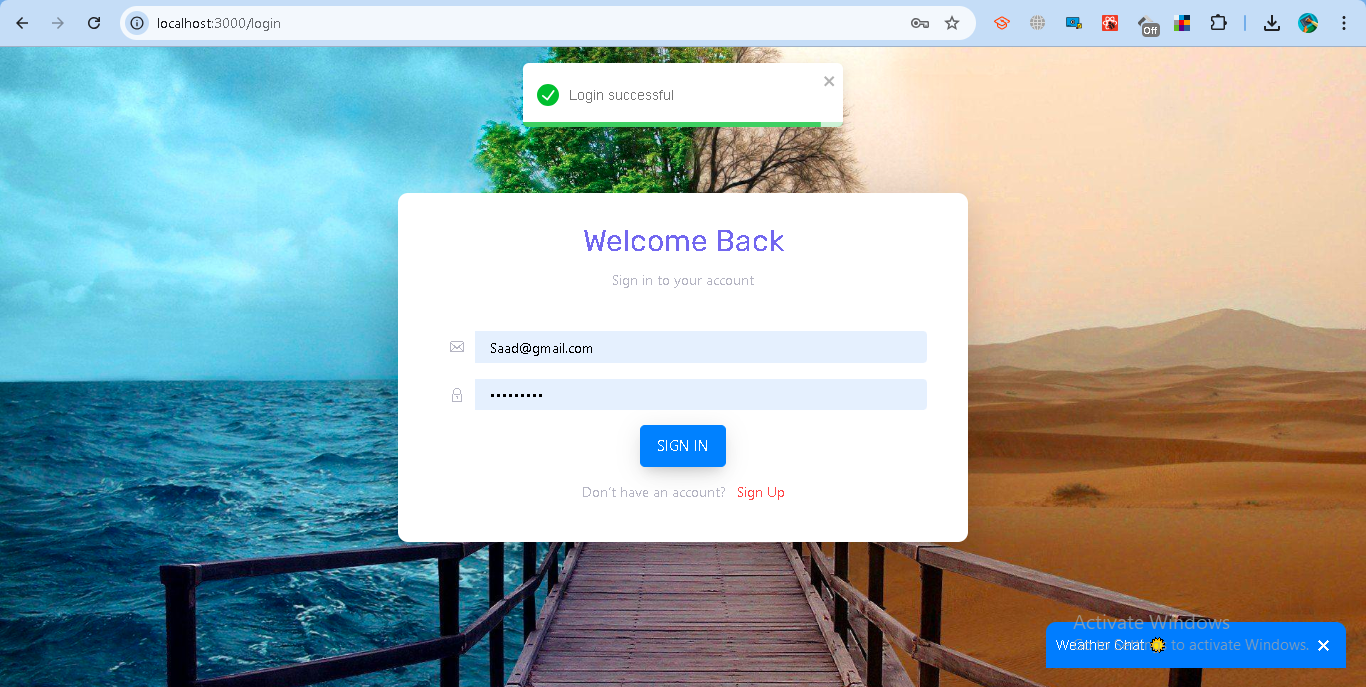
# **1. Project Report**

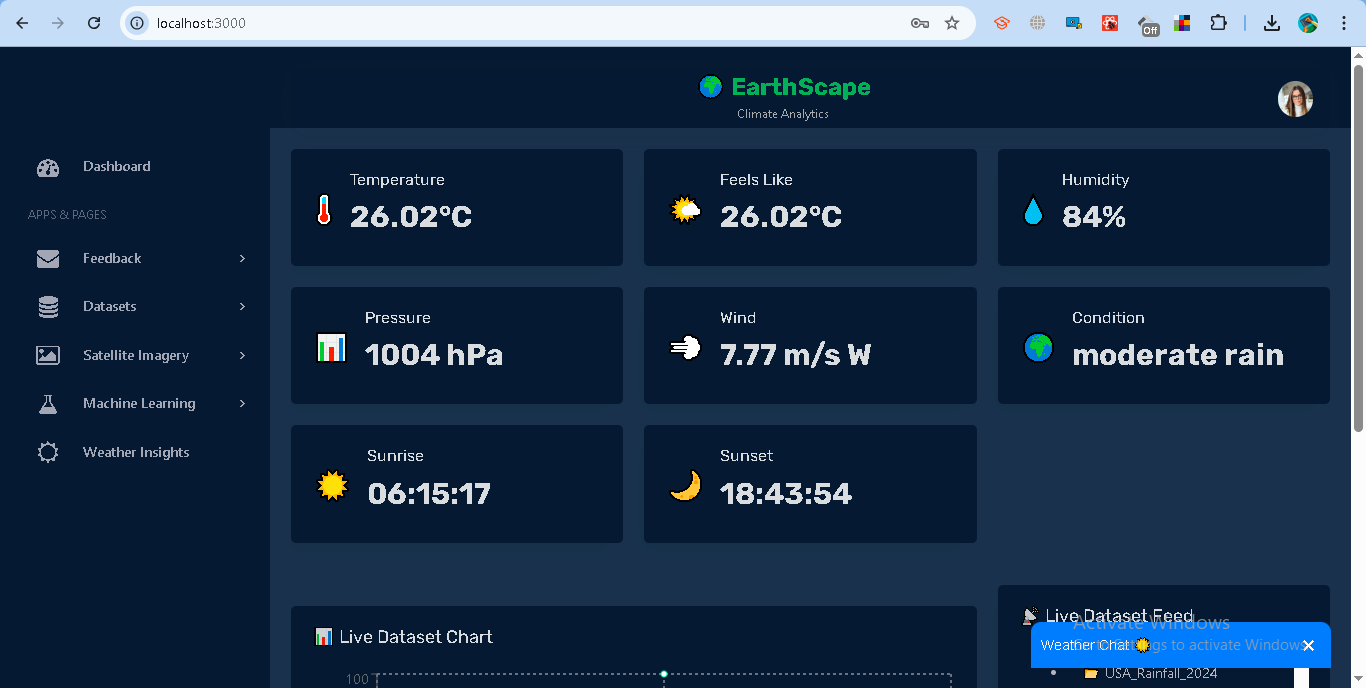
## **1.1 Authentication and Authorization**

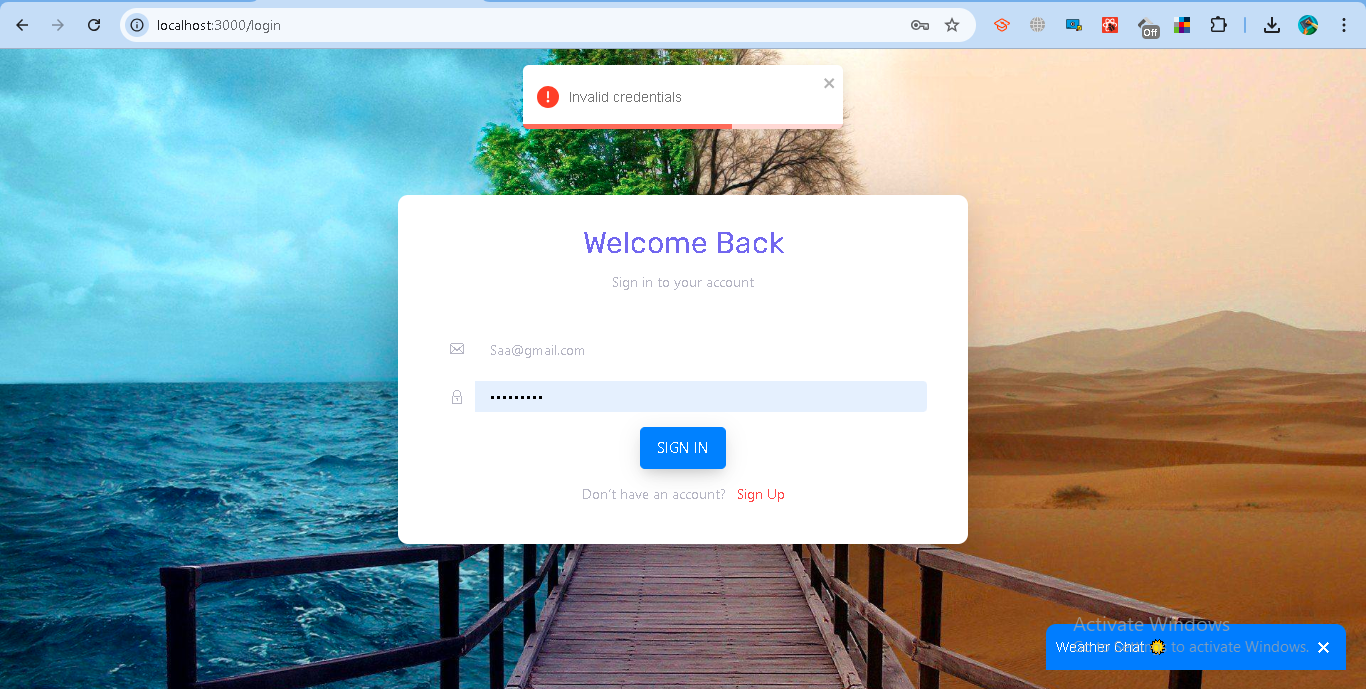
### **Registering as a User (Analyst)**



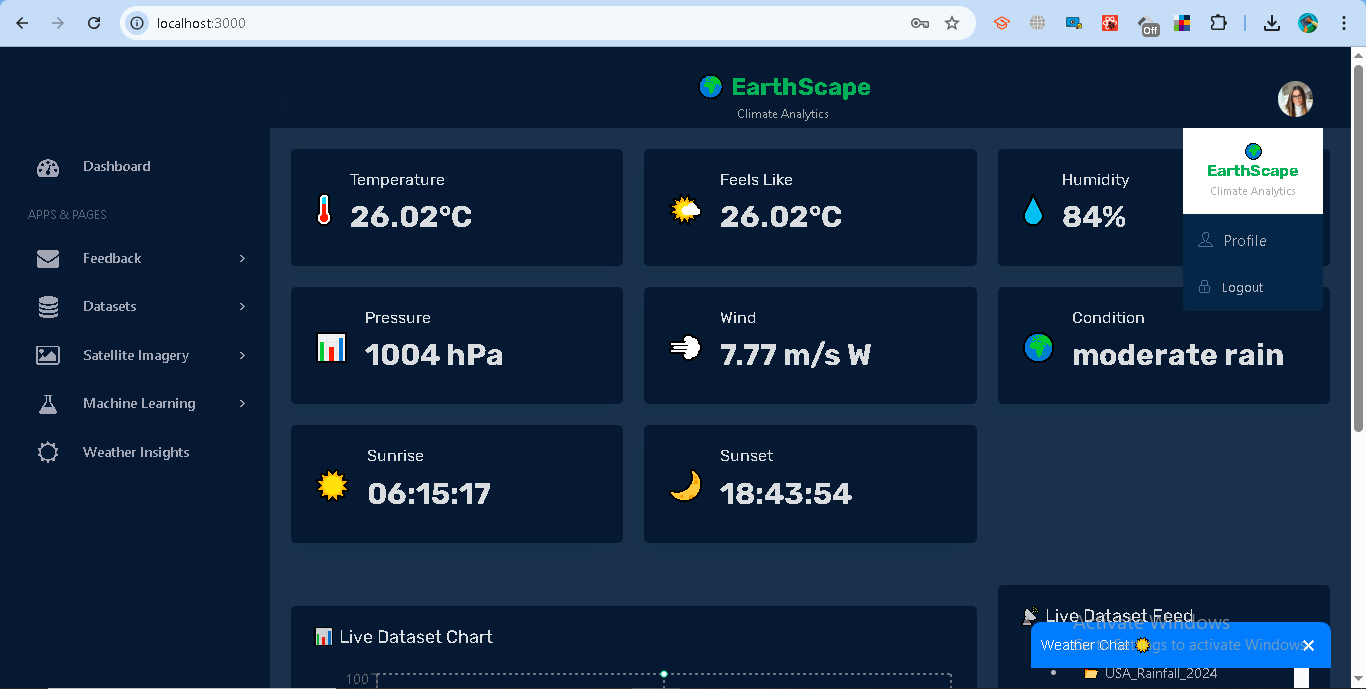
### **Logging In**



* 



### **Logging Out**

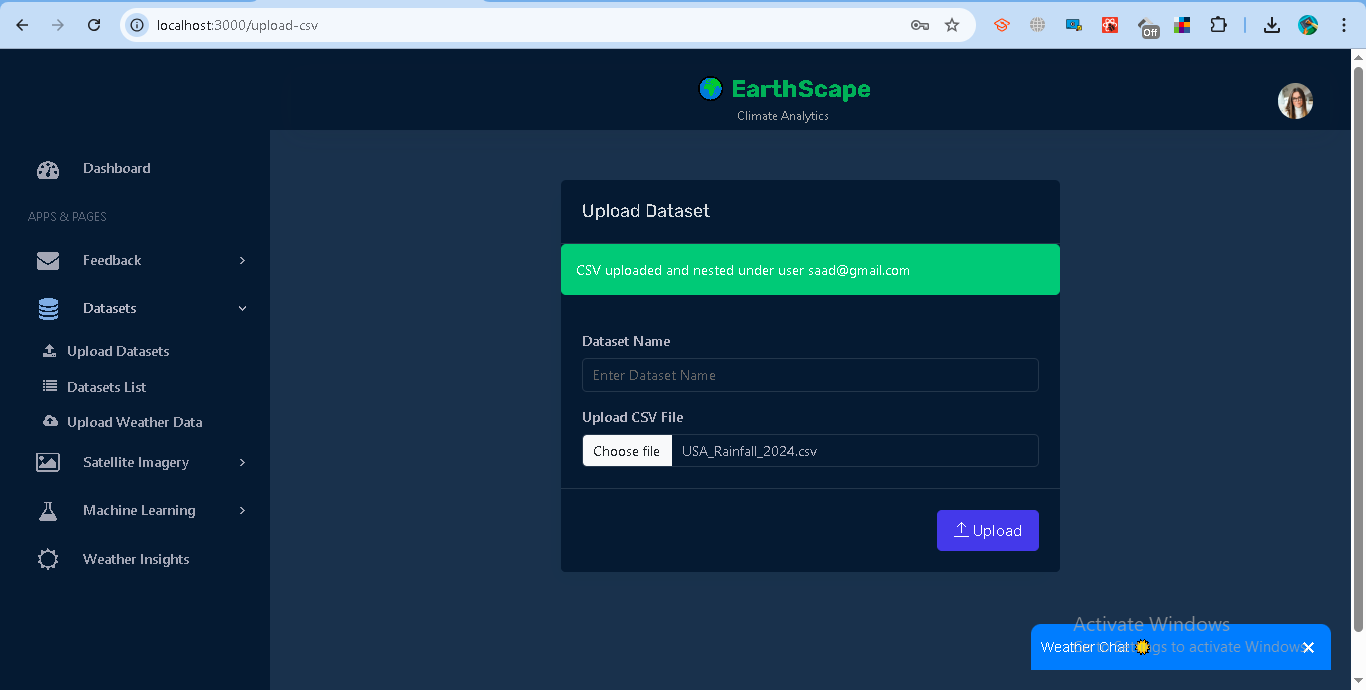


# **1.2 Data Ingestion**

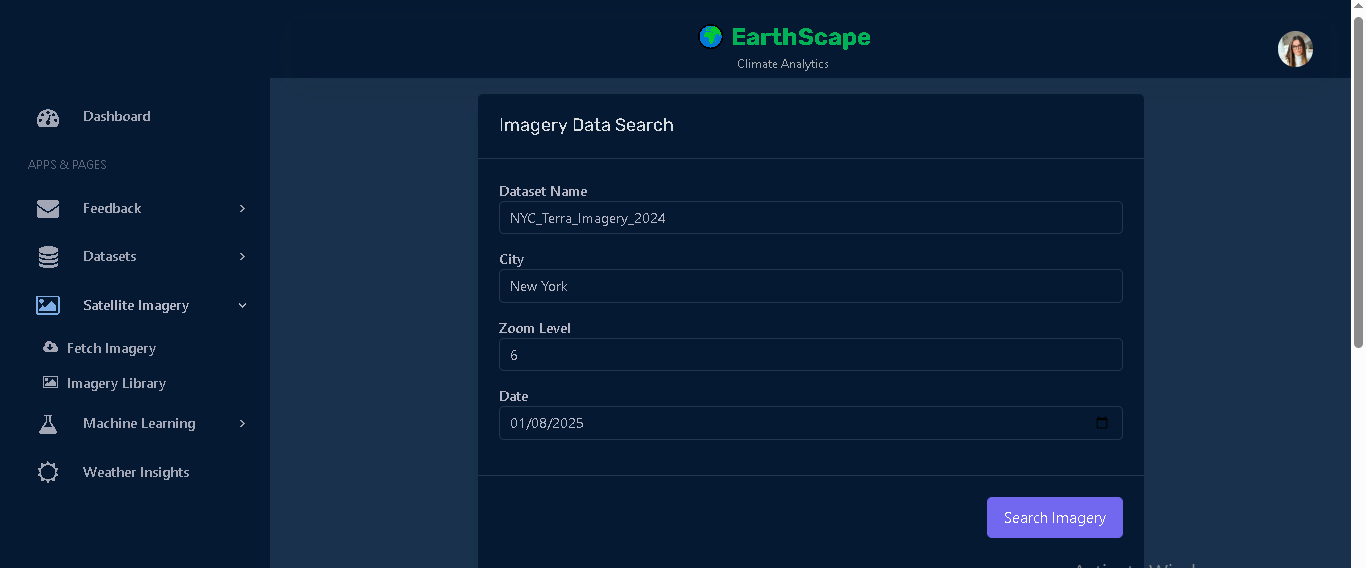
### **Purpose**

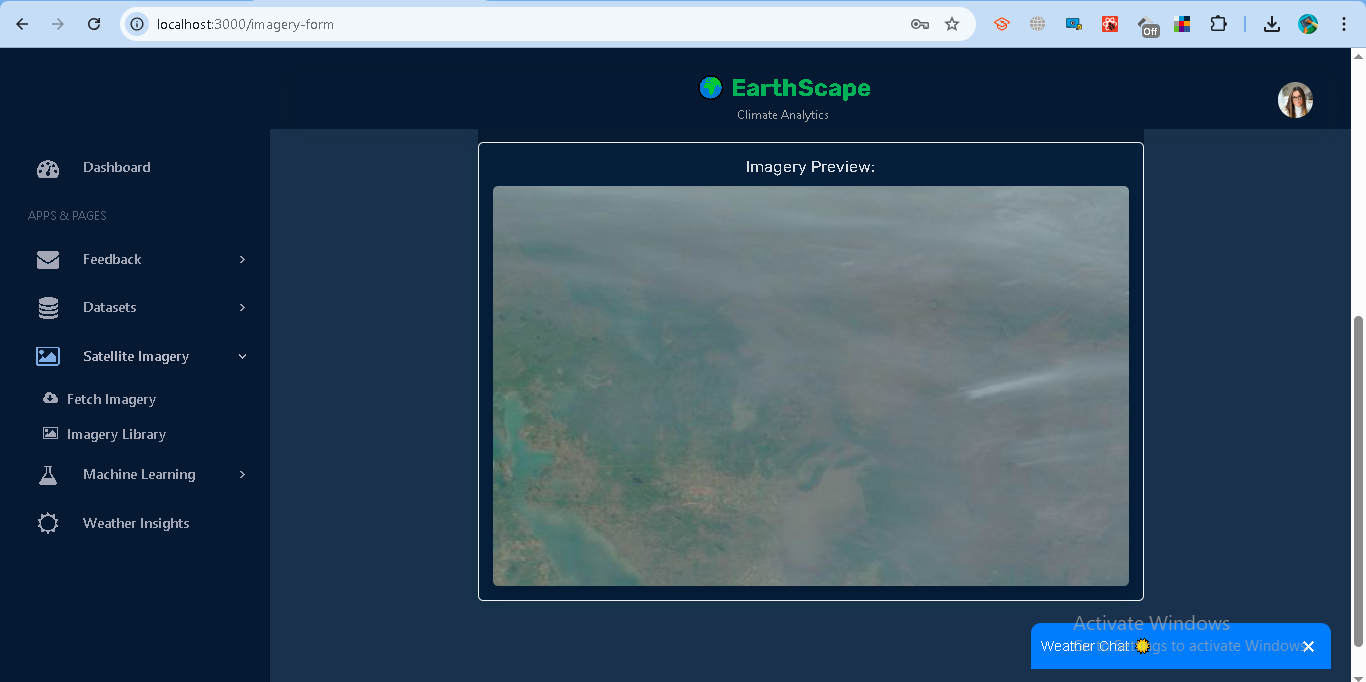
The data ingestion module allows users to upload and manage climate-related datasets such as CSV files, satellite imagery, and weather records. This ensures that both historical and real-time data sources can be stored, processed, and later analyzed.



**

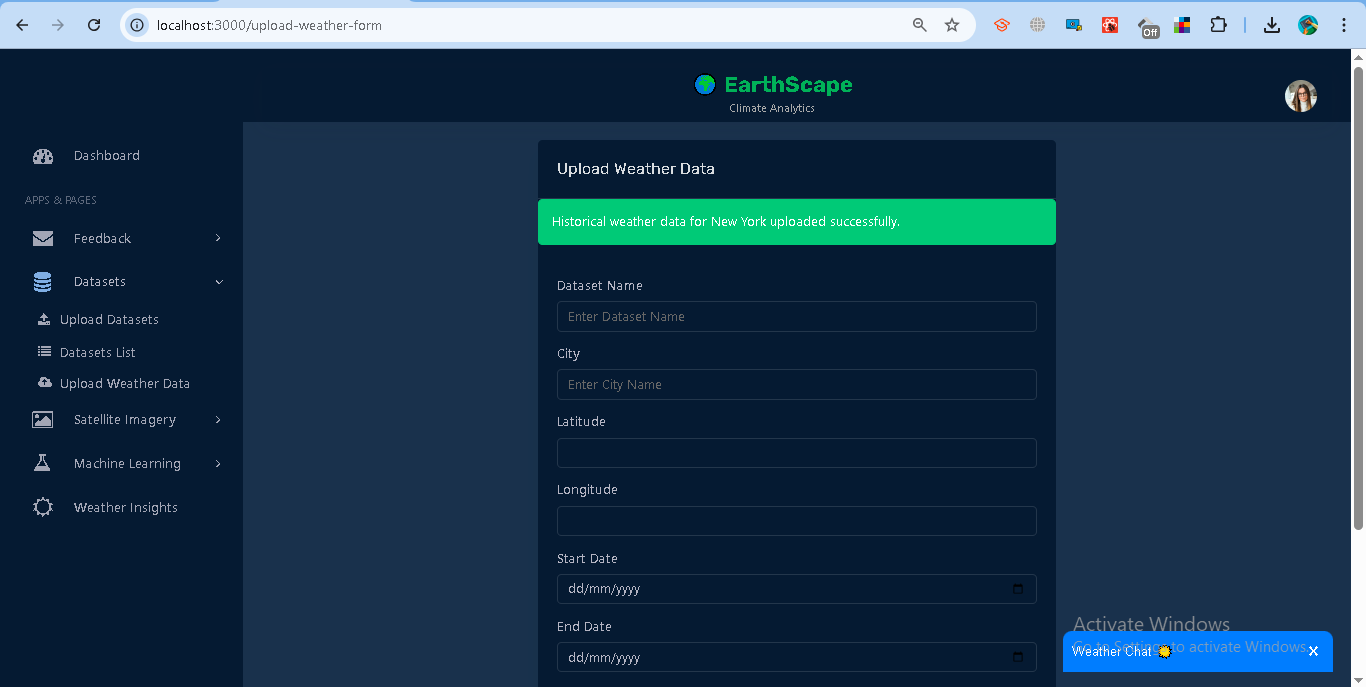
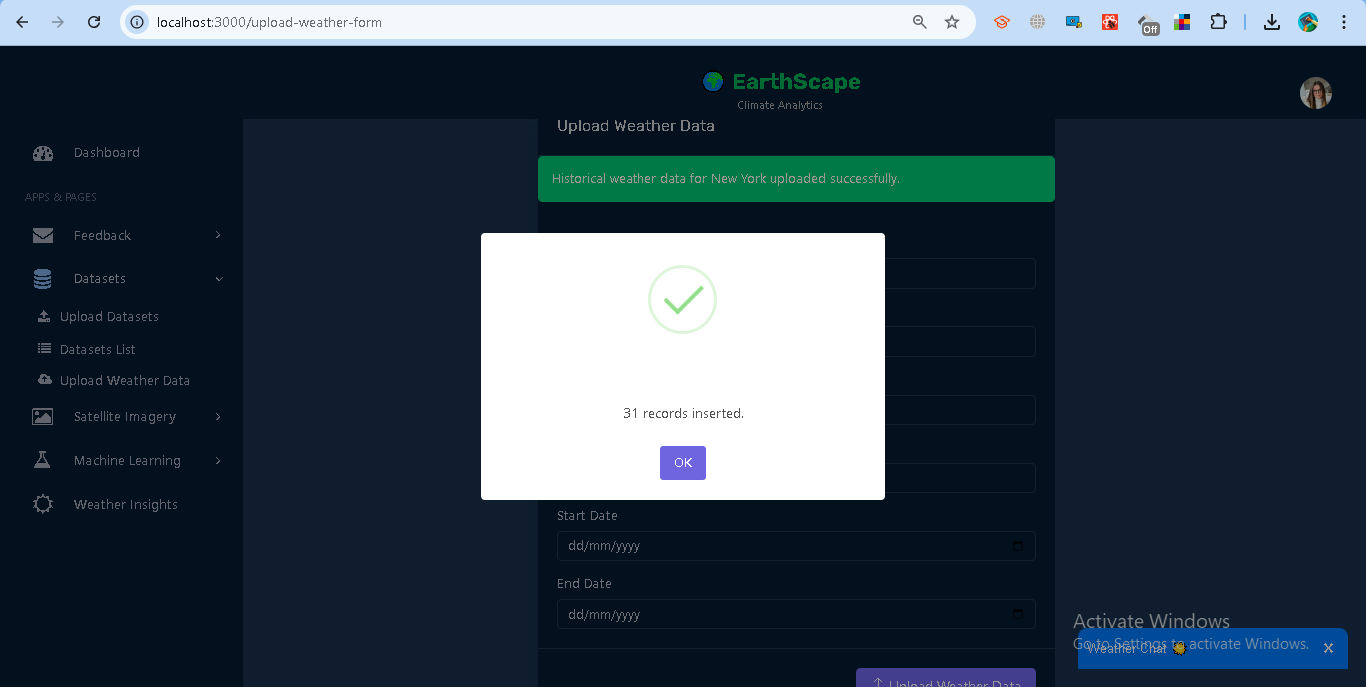
#### **B. Uploading Satellite Imagery**





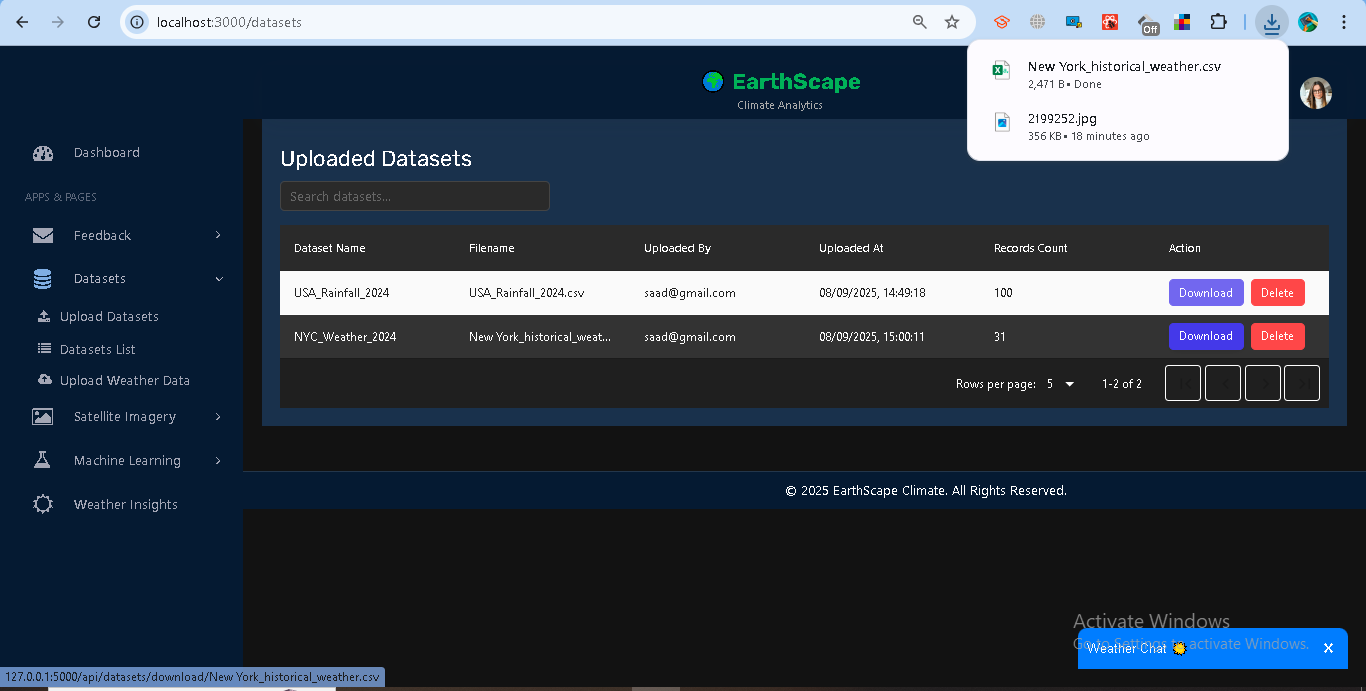
#### **C. Uploading Weather Data**



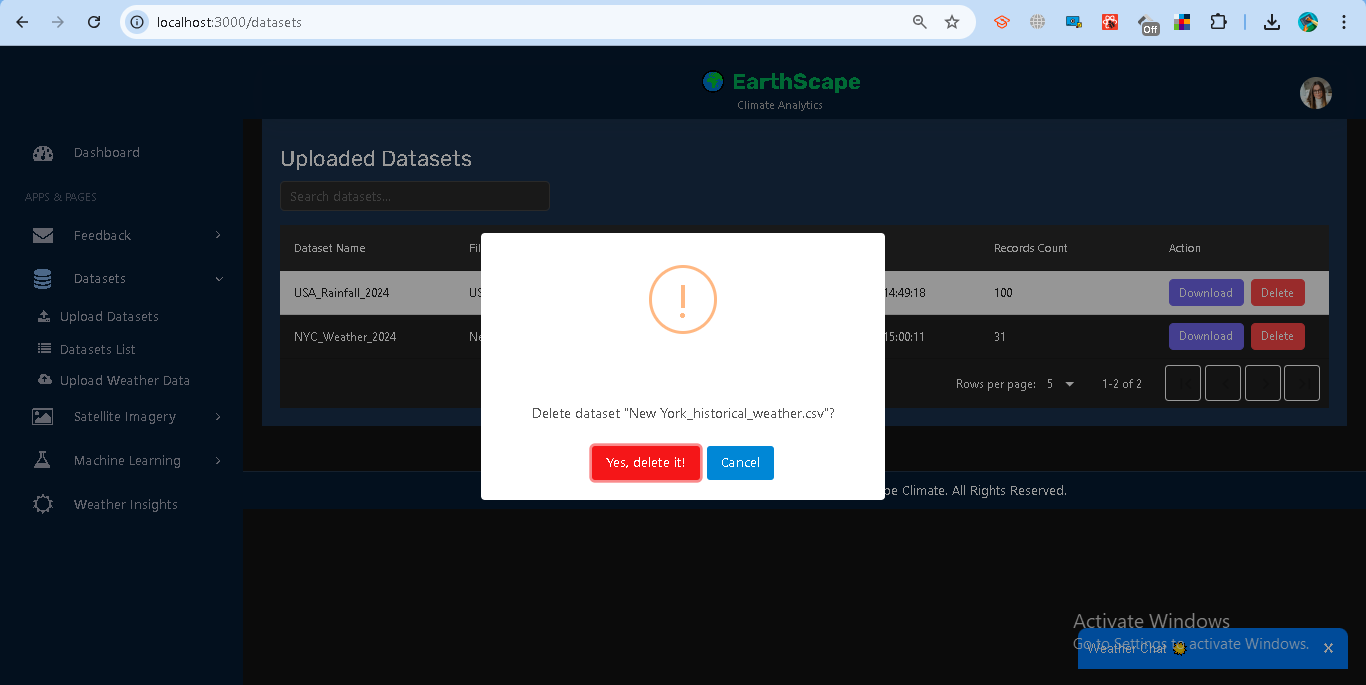


### **Dataset Management**

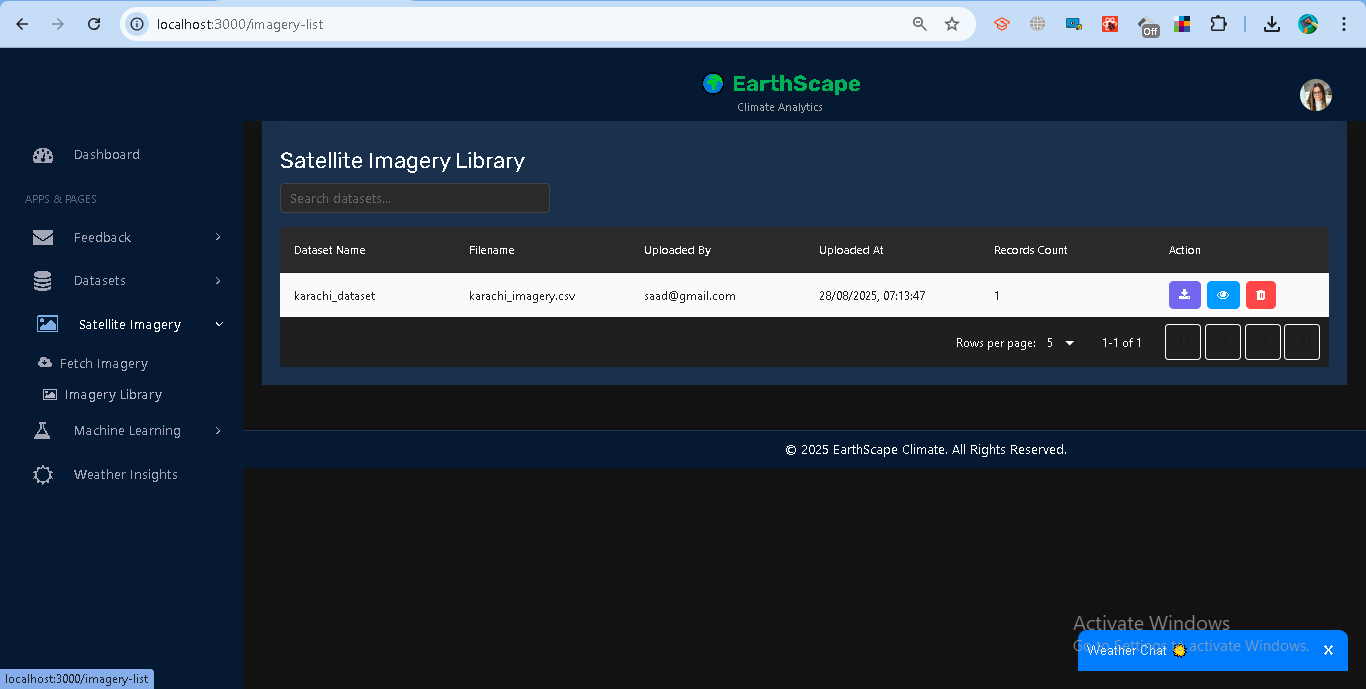
* **Download:** Each dataset provides a direct download link for .csv files.

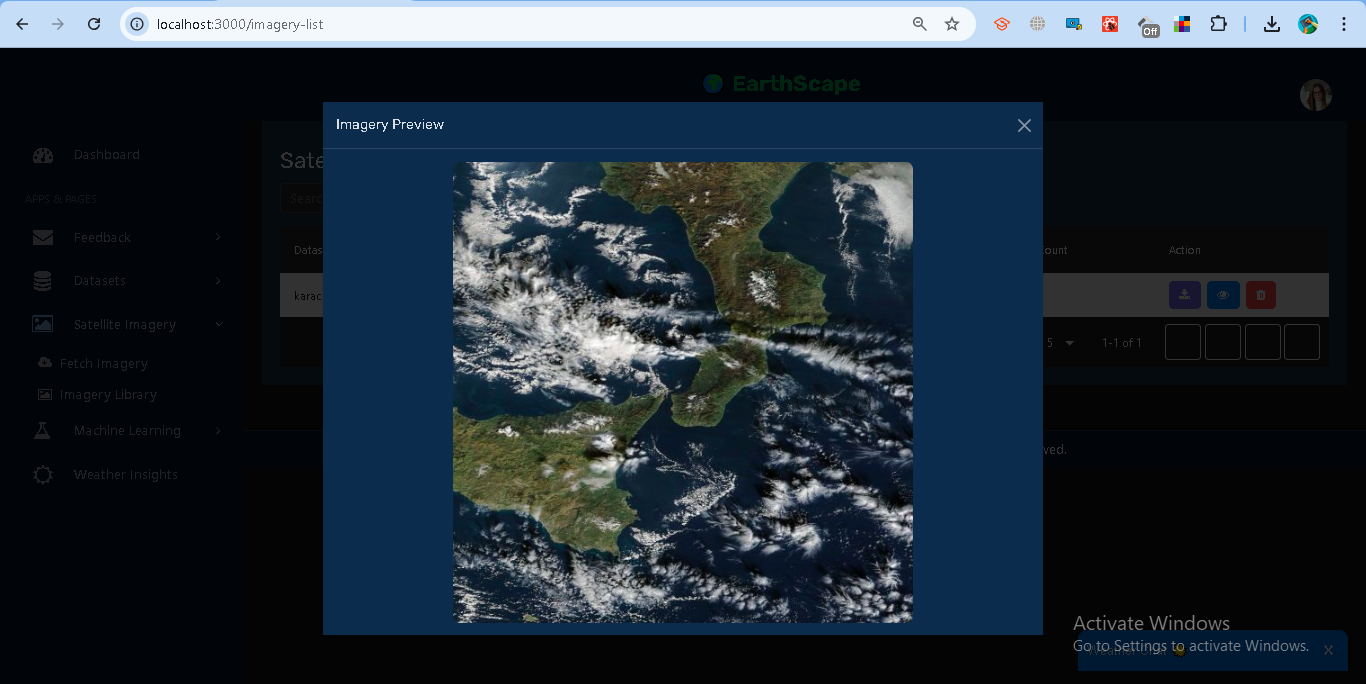


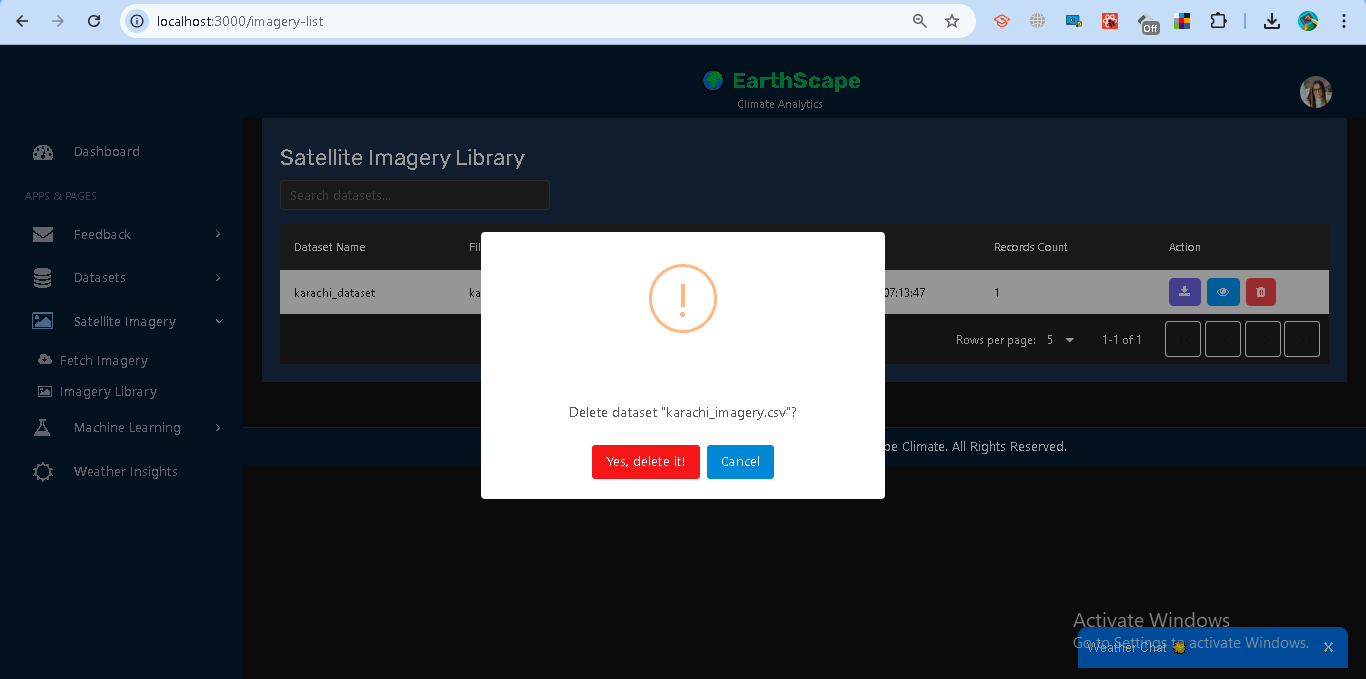
* **Delete:** Users may delete their own datasets; admins can delete all datasets.



* **View Datasets:** All uploaded datasets can be viewed under **Satellite Imagery → *Imagery Library*.**

****

****

****

## **1.3 Data Storage**

### **Implementation**

Instead of using Hadoop Distributed File System (HDFS), this project leverages **MongoDB Atlas** for storing climate-related datasets in a **scalable and fault-tolerant** way. MongoDB’s document-oriented design provides flexibility for handling diverse climate datasets such as CSV uploads, satellite imagery metadata, and weather station records.

* **Datasets Collection (datasets)**
  + Stores uploaded CSV datasets along with metadata (dataset name, file path, uploader, records, record counts).

Example structure:  
  
 {

"user\_id": "65ab3e...",

"datasets": [

{

"dataset\_name": "NYC\_Weather\_2024",

"filename": "nyc\_weather.csv",

"file\_path": "/dataset/nyc\_weather.csv",

"uploaded\_at": "2025-01-12T08:30:00Z",

"uploaded\_by": "admin@example.com",

"records\_count": 100,

"records": [ { "date": "2024-01-01", "temperature": 5, "humidity": 80 } ]

}

]

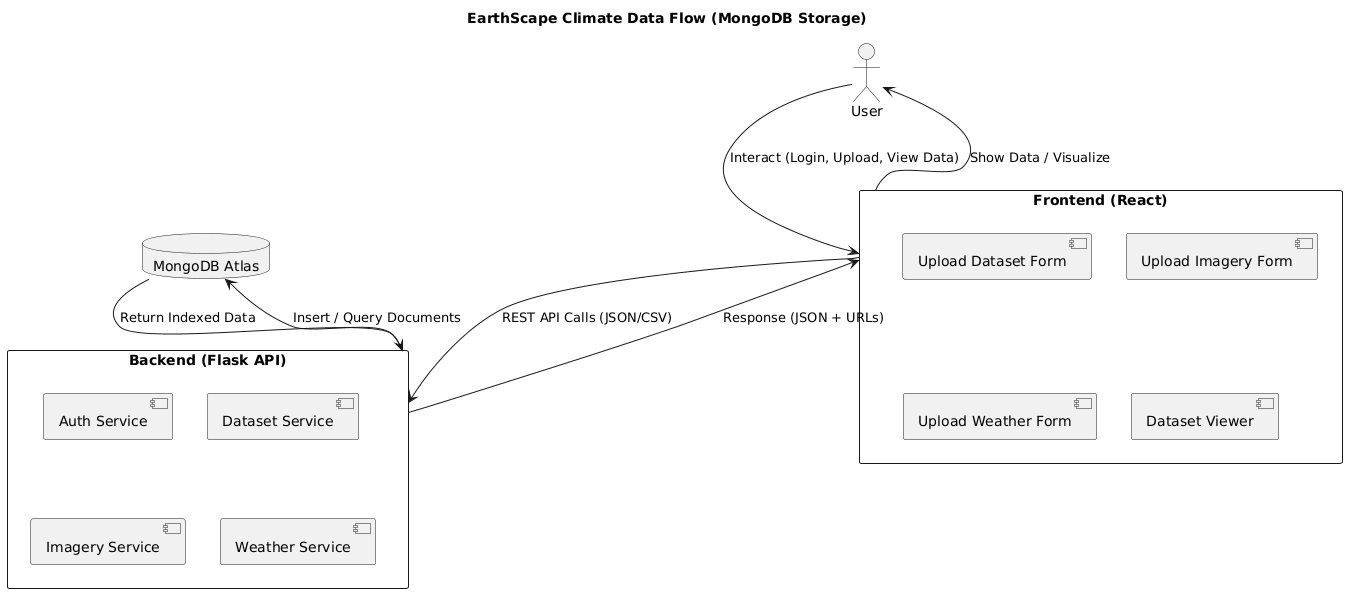
}

* **Satellite Imagery Collection (satellite\_imagery)**
  + Stores metadata about imagery requests (city, lat/lon, zoom level, NASA GIBS URL, generated CSV file, etc.).
* **User Profile Collection (users)**
  + Stores user credentials, roles, and cities, enabling role-based access control.

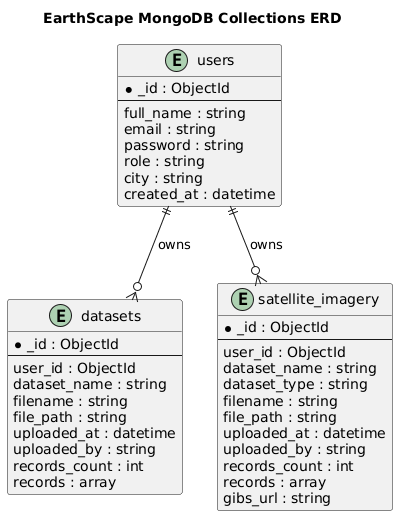
### **Optimization Strategies**

* **Indexing**
  + MongoDB indexes are created on fields like user\_id, datasets.filename, and satellite\_imagery.dataset\_name to speed up queries.
* **Partitioning / Sharding**
  + MongoDB supports **sharding** for horizontal scaling when datasets grow large. This ensures queries and storage are distributed across multiple nodes.
* **Document Nesting**
  + Datasets are stored as **nested arrays under each user document**, ensuring quick retrieval of all datasets for a given user.
* **File Storage**
  + Uploaded CSVs and imagery metadata are stored locally under /dataset and referenced in MongoDB with **download URLs** for retrieval.

**Data Flow**

****

**ERD**

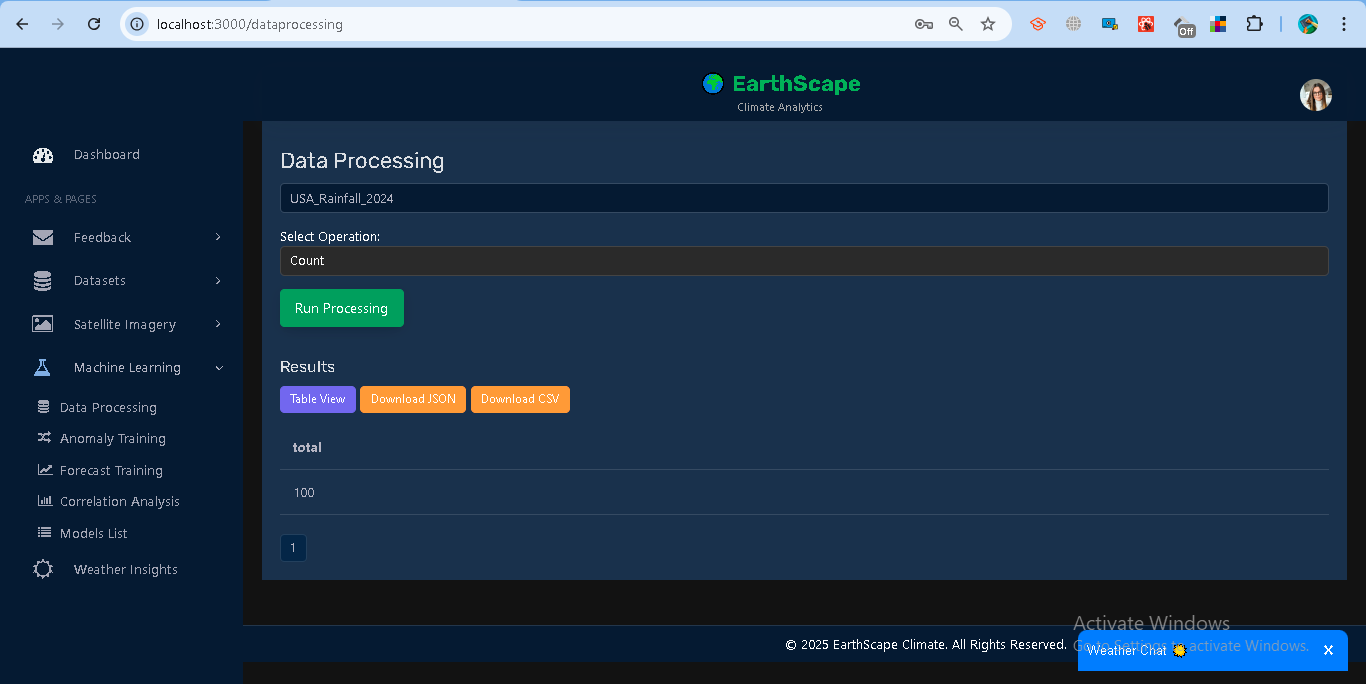
****

### **1.4 Data Processing**

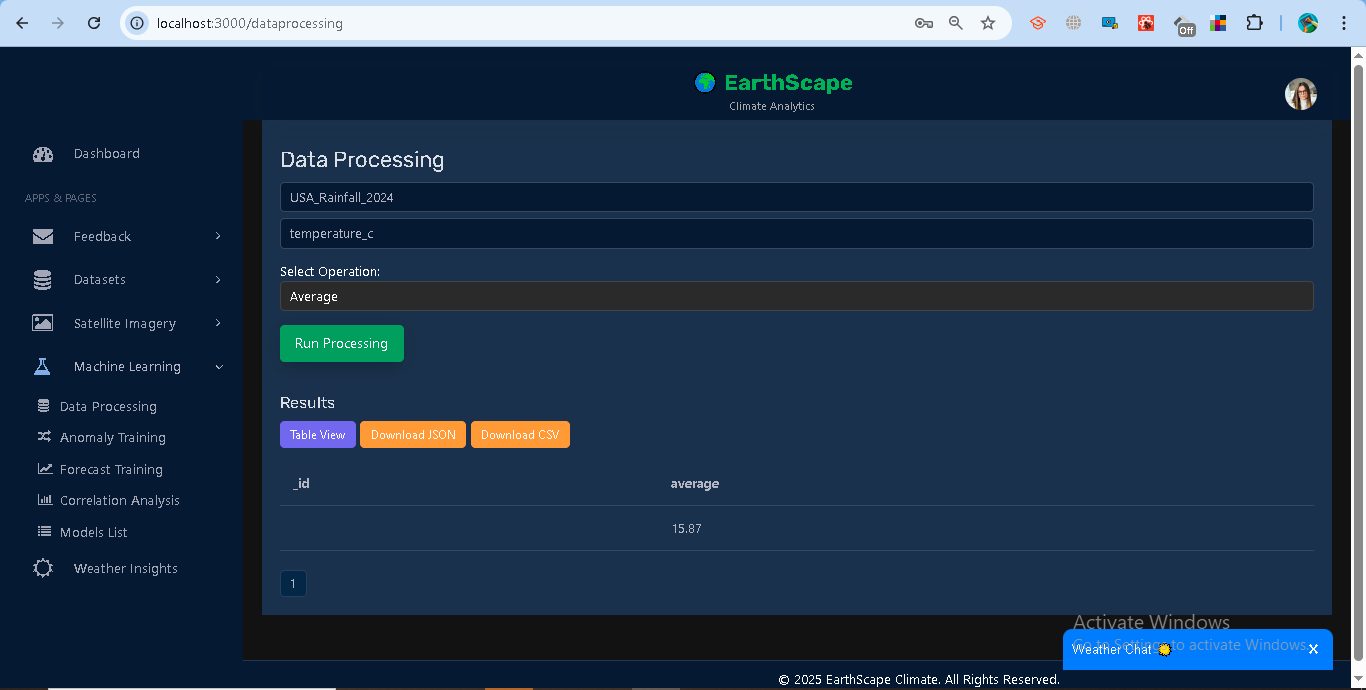
**Purpose** Allows users to analyze climate datasets (CSV, Weather, Satellite) stored in MongoDB. Operations include record counts, averages, groupings, and simple temporal pattern detection.

**Available Operations**

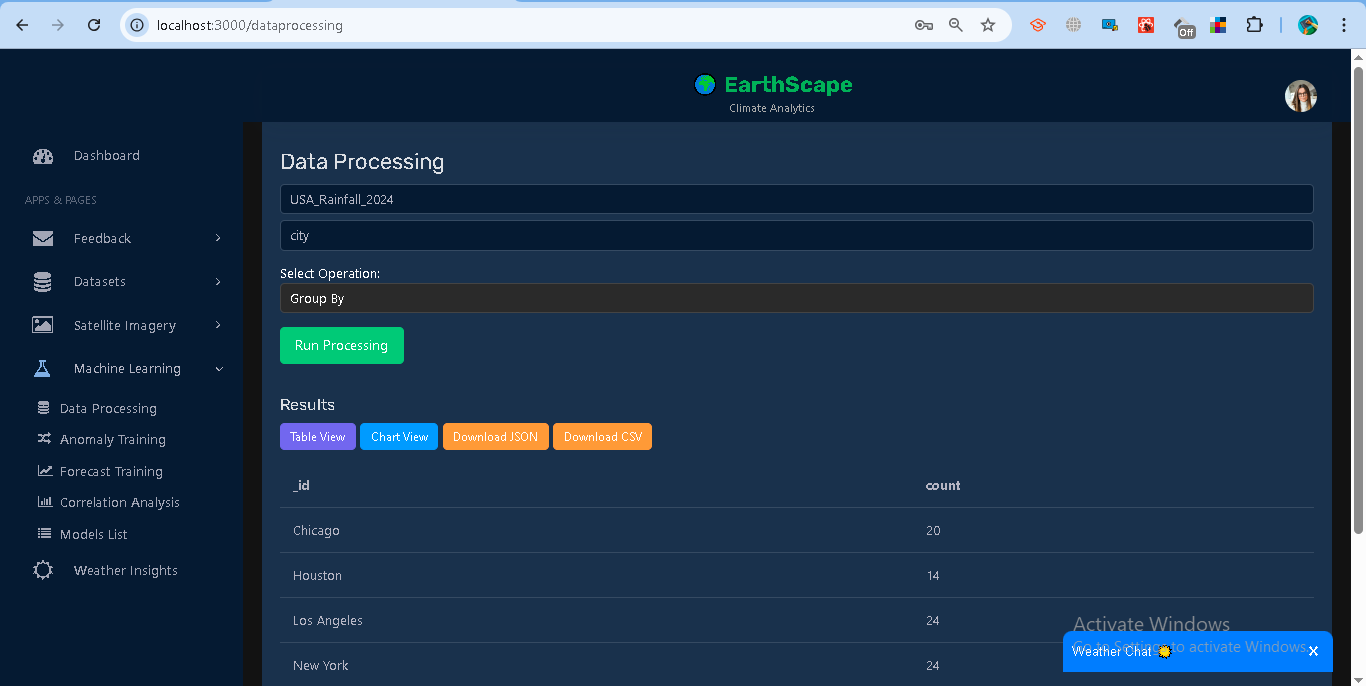
* **Count** → total records.



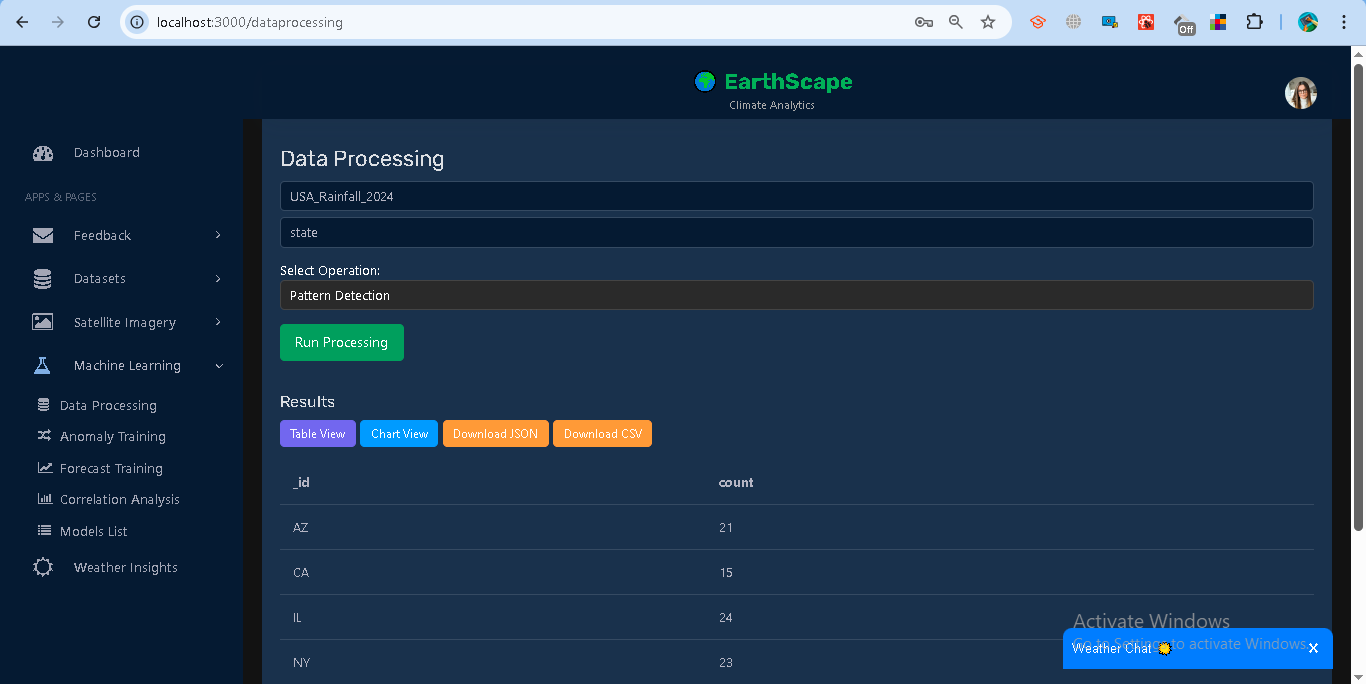
* **Average** → mean of numeric field.

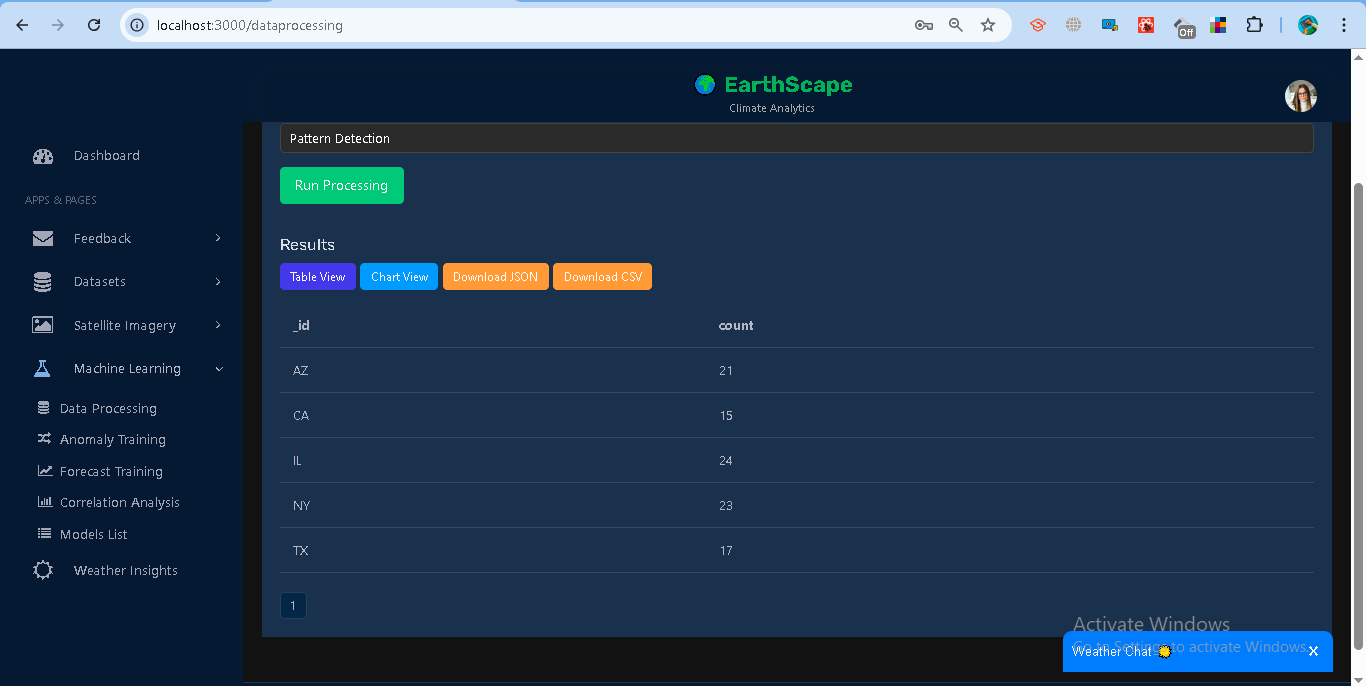


* **Group By** → group by field (city, value bins).

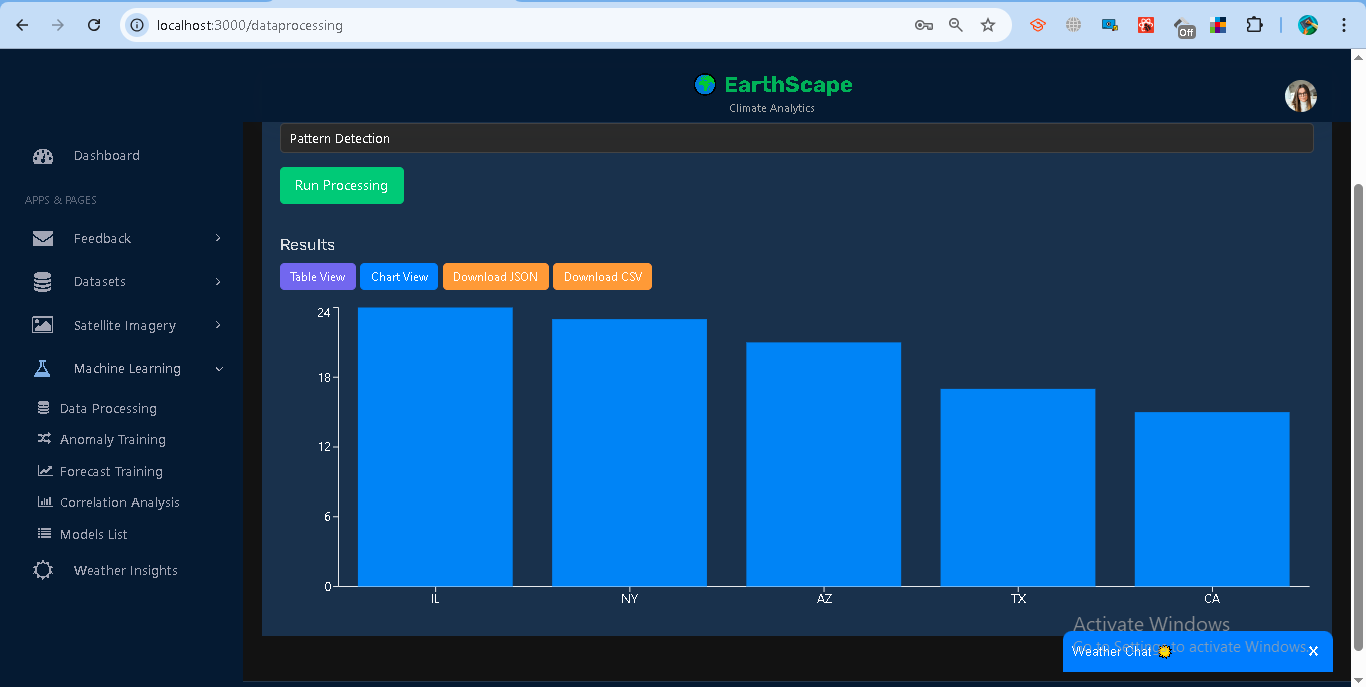


* **Pattern** → detect seasonality (group by month if date).





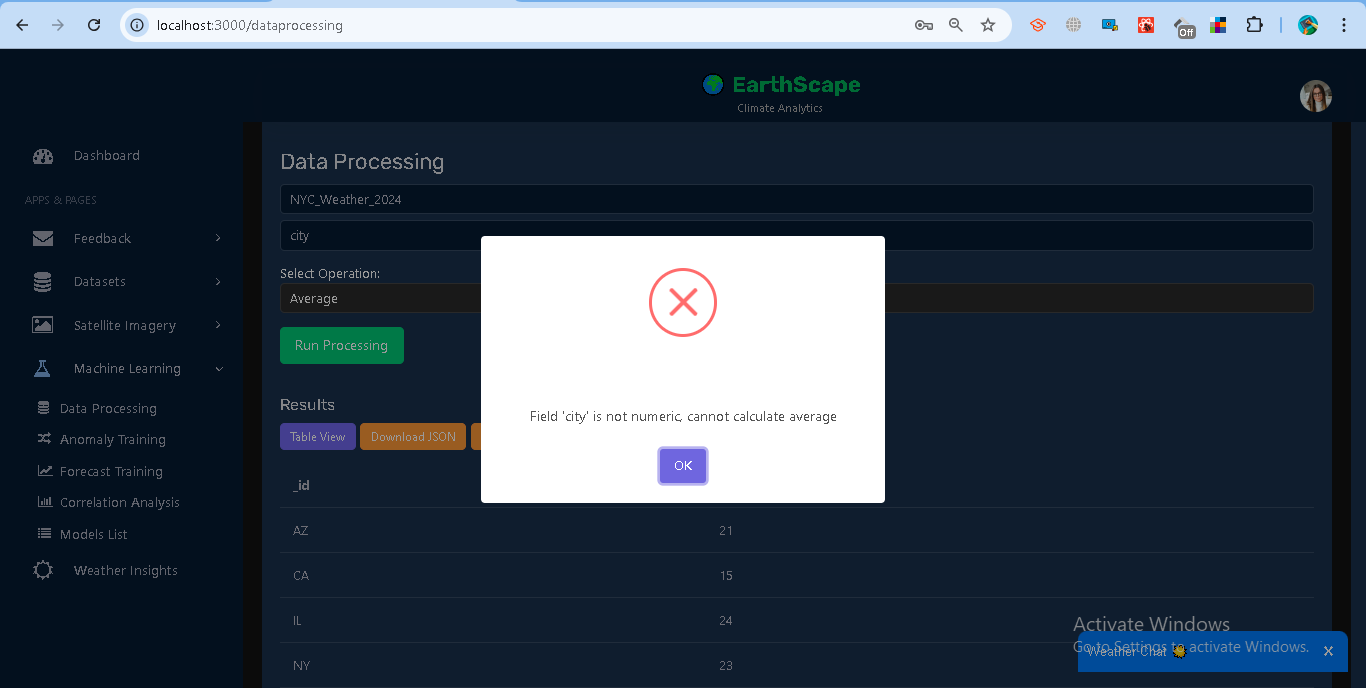
1. View results as **Table** (with pagination) or **Chart** (Bar/Line).



1. Download results as **JSON** or **CSV**.

**Examples**

* Count on NYC\_Weather\_2024 → { total: 1000 }
* Average on temperature → { average: 17.42 }
* Group By city → [{ \_id: "New York", count: 123 }, ...]
* Pattern on datetime → monthly counts { \_id: 1, count: 34 }



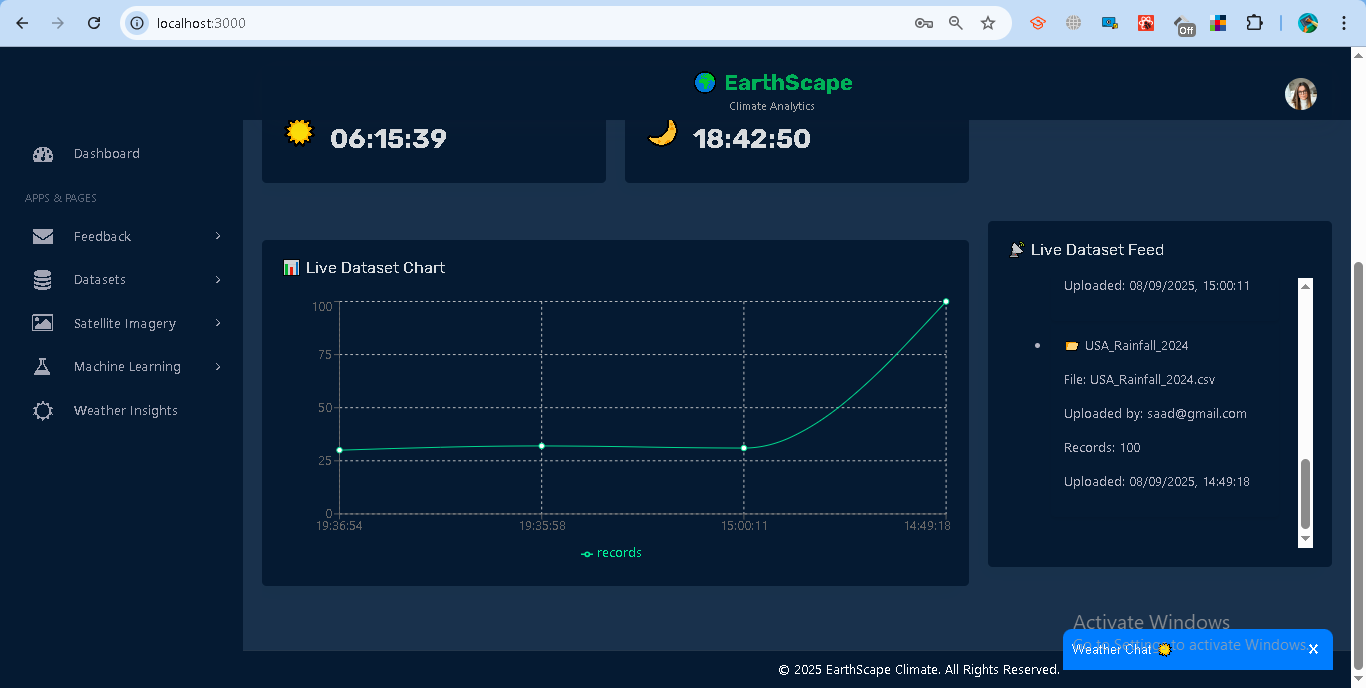
## **1.5 Real-time Data Processing**

The system integrates **real-time data streaming** with **batch data** for complete analysis. Users can view live updates on datasets and temperature records without refreshing the page.

### **Steps**

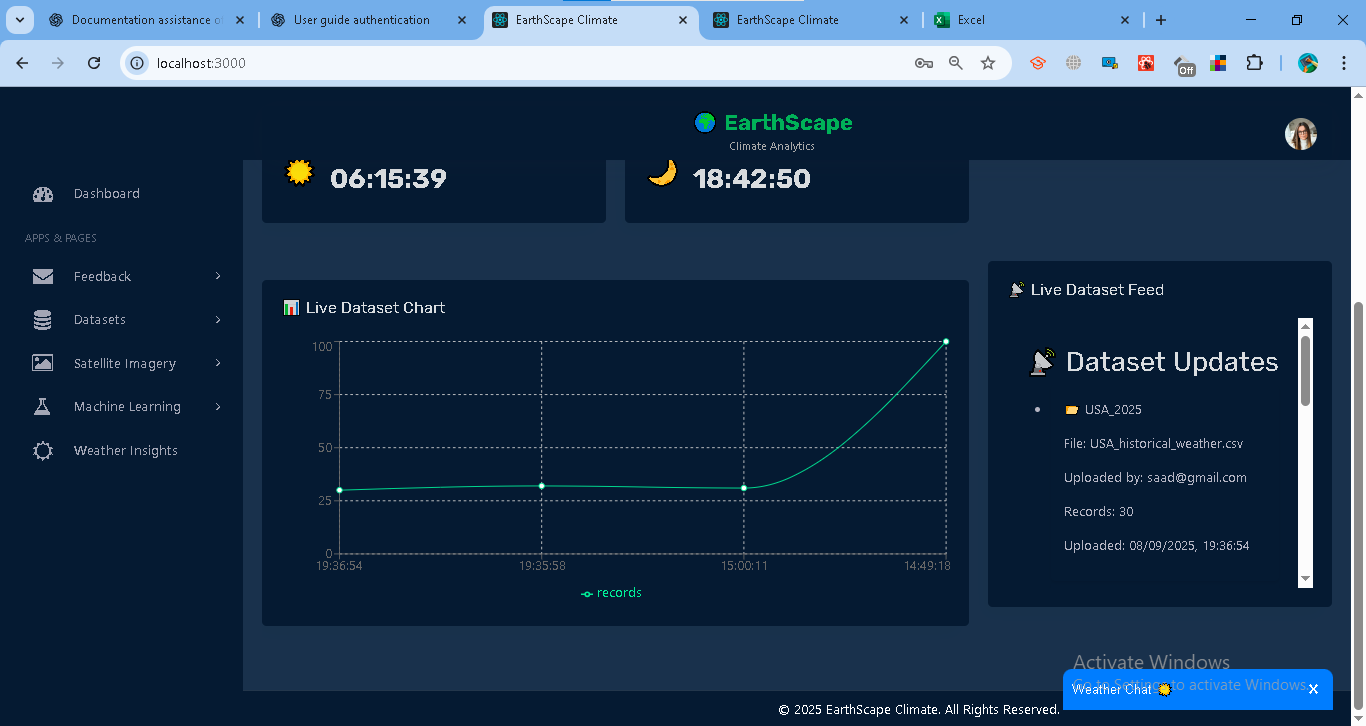
**Live Dataset Chart**

* Shows number of records over time.
* Loads historical data from /datasets/livefeed.
* Subscribes to /stream/datasets?token=... for new updates.



**Live Dataset Feed**

* Lists the latest 10 datasets.
* Displays file, uploader, record count, and timestamp.
* Option to expand and view sample records.



**Live Temperature (per City)**

* Plots temperature trends across multiple cities.
* Loads recent records from /datasets/latest?token=....
* Streams updates via /stream/datasets?token=....
* Cities can be selected from a dropdown.



### **Access Control**

* Analysts: see only their datasets.
* Admins: see datasets for the owner of the latest upload.

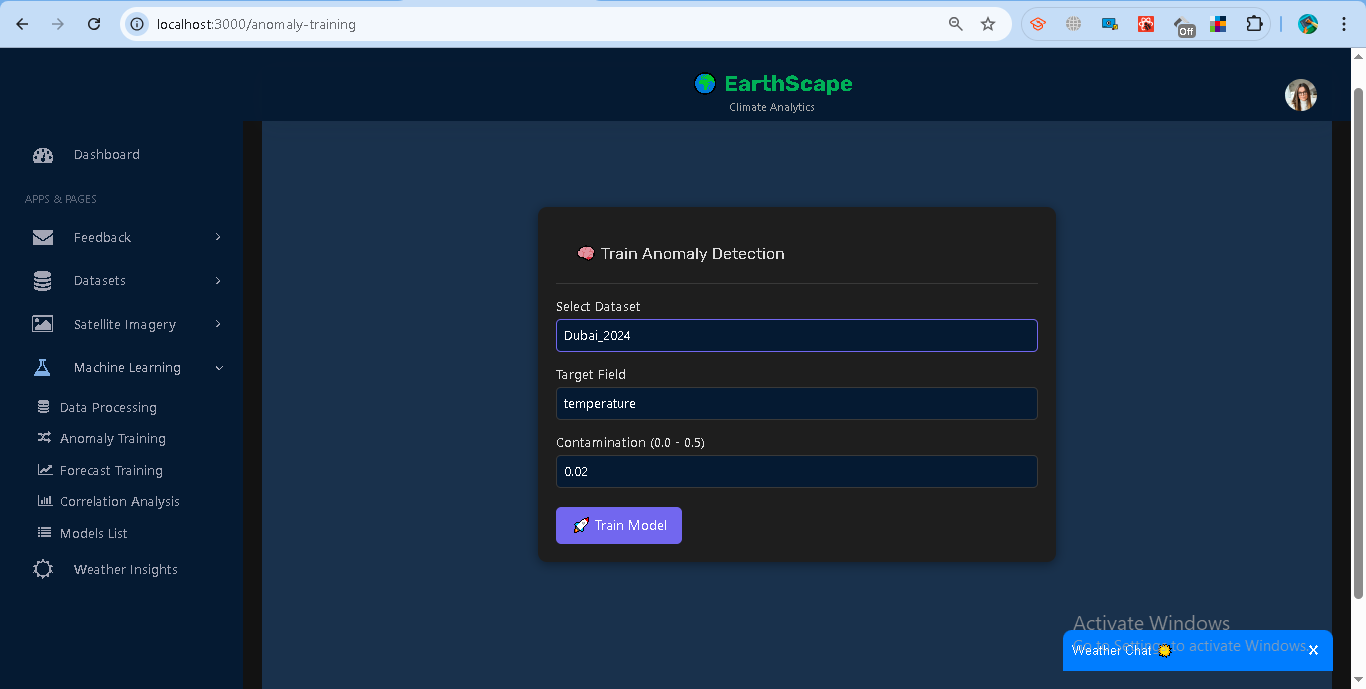
### **Error Handling**

* **401 Unauthorized** → Invalid or expired token.
* **404 No dataset found** → No datasets uploaded yet.
* **No updates** → Check SSE connection, token, and database stream settings.

## **1.6 Anomaly Training**

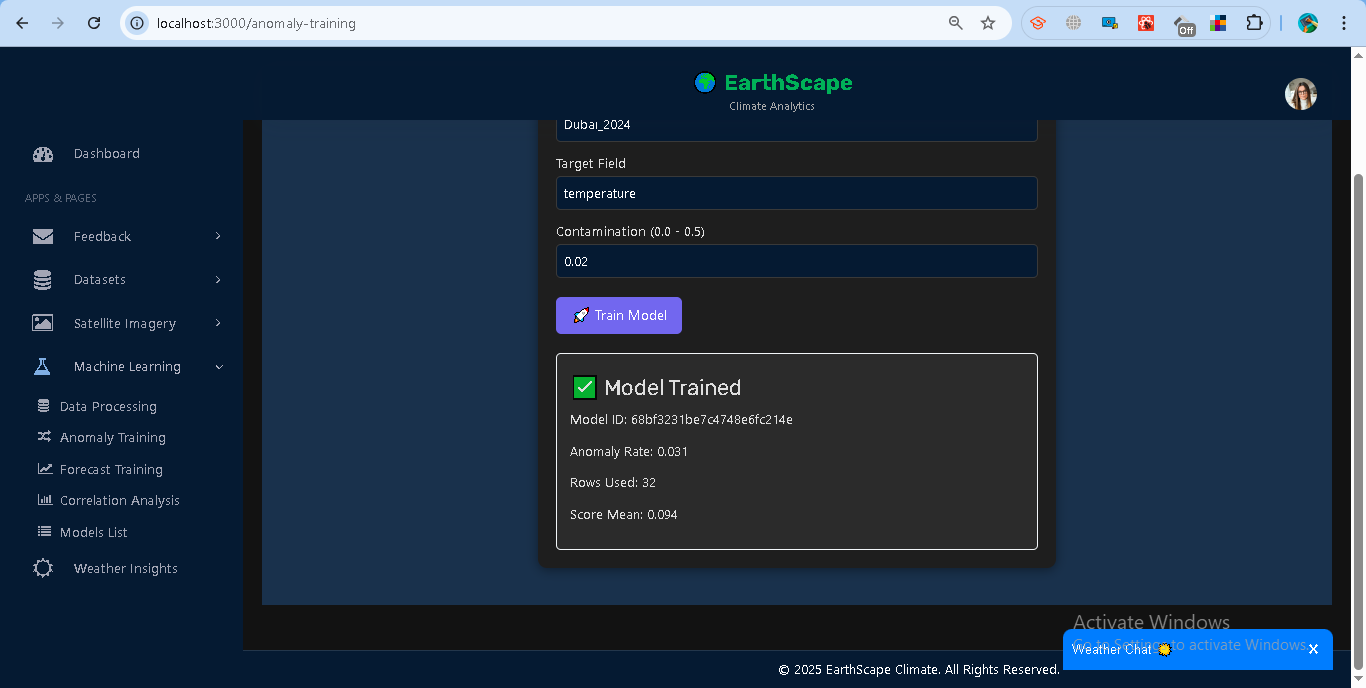
The system provides anomaly detection using the **Isolation Forest model**. This helps identify unusual climate values (e.g., abnormal temperatures) in uploaded datasets.

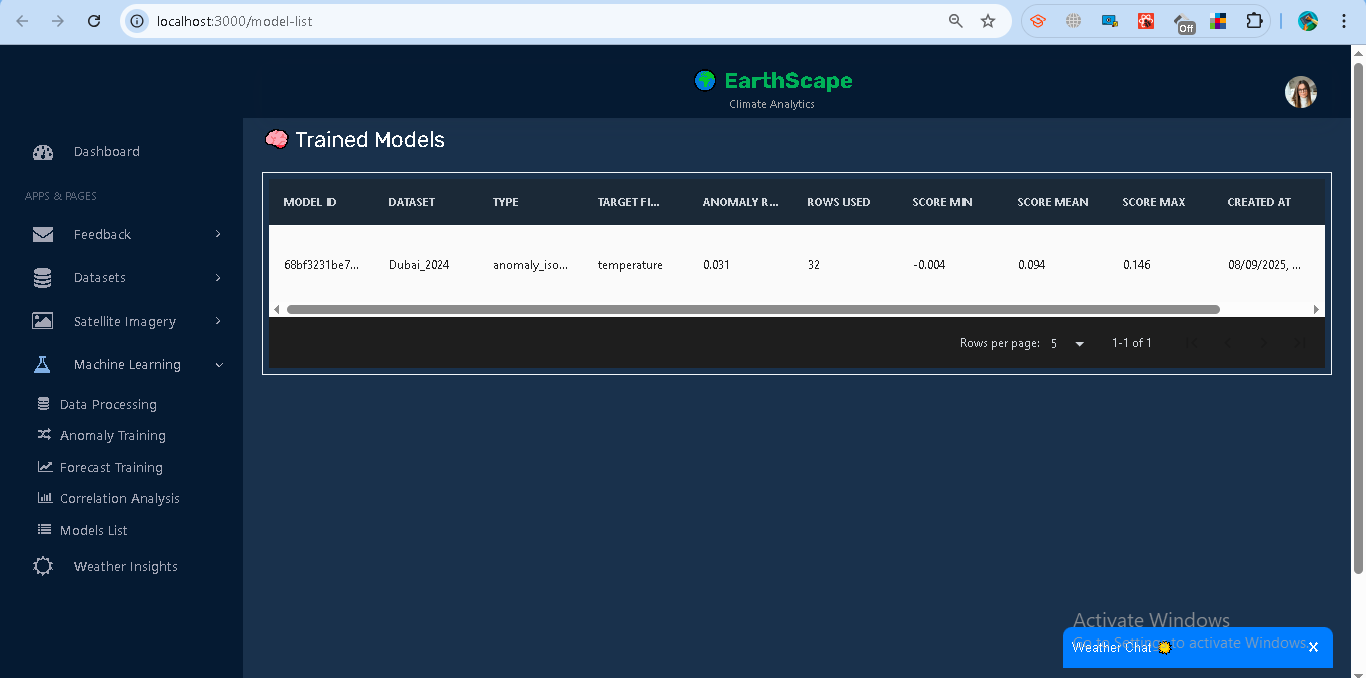
### **Steps**

**Train a Model**

**Results**

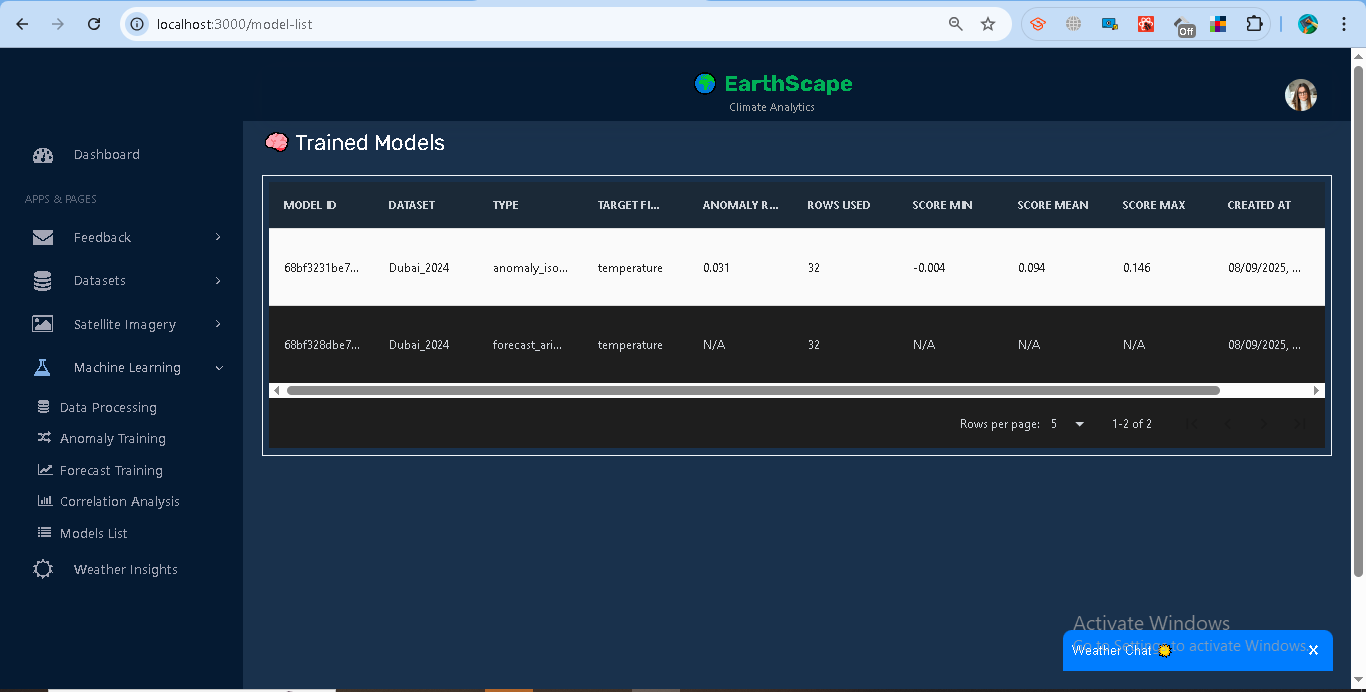
* Model ID is generated and stored.
* Training summary shows:  
  + **Anomaly Rate** → percentage of detected anomalies.
  + **Rows Used** → records used in training.
  + **Score Mean** → average decision score.





## **1.7 Forecast Training**

The system supports **time-series forecasting** using the **ARIMA model**. This allows predicting future values (e.g., temperature trends) from uploaded datasets.



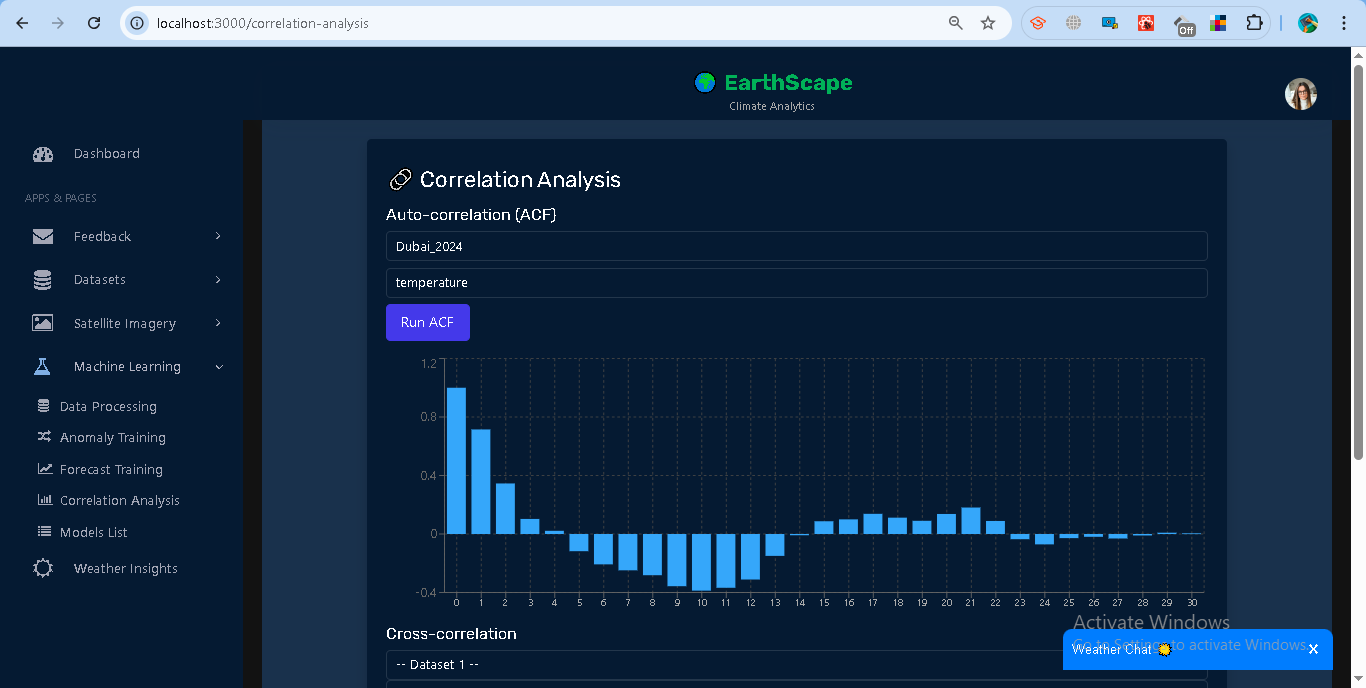
## **1.8 Correlation Analysis**

The system provides correlation tools to help understand relationships within and between climate datasets. Analysts can compute **Auto-correlation (ACF)**, **Cross-correlation**, and **Correlation Matrices**.

### **Features**

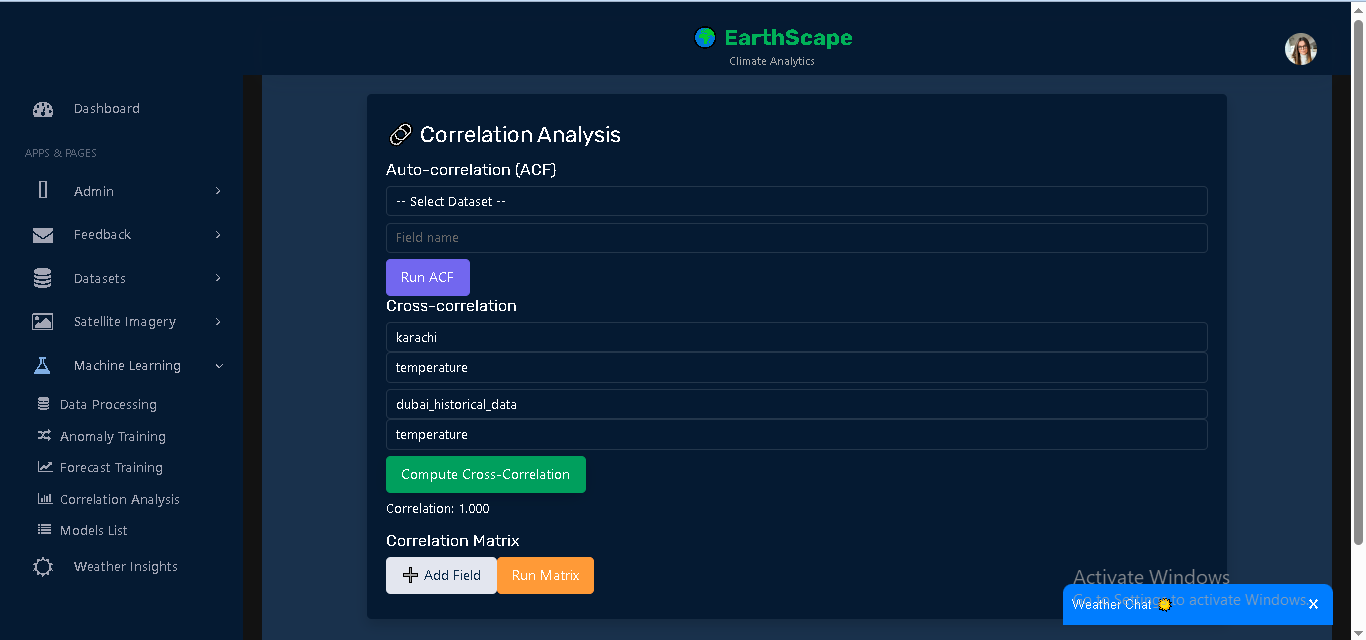
**Auto-correlation (ACF)**

* Measures how a field correlates with its past values (lags).
* Steps:  
  1. Select a dataset.
  2. Enter a **Target Field** (numeric column, e.g., *temperature*).
  3. Run ACF.
* Results: Bar chart of lag vs correlation.



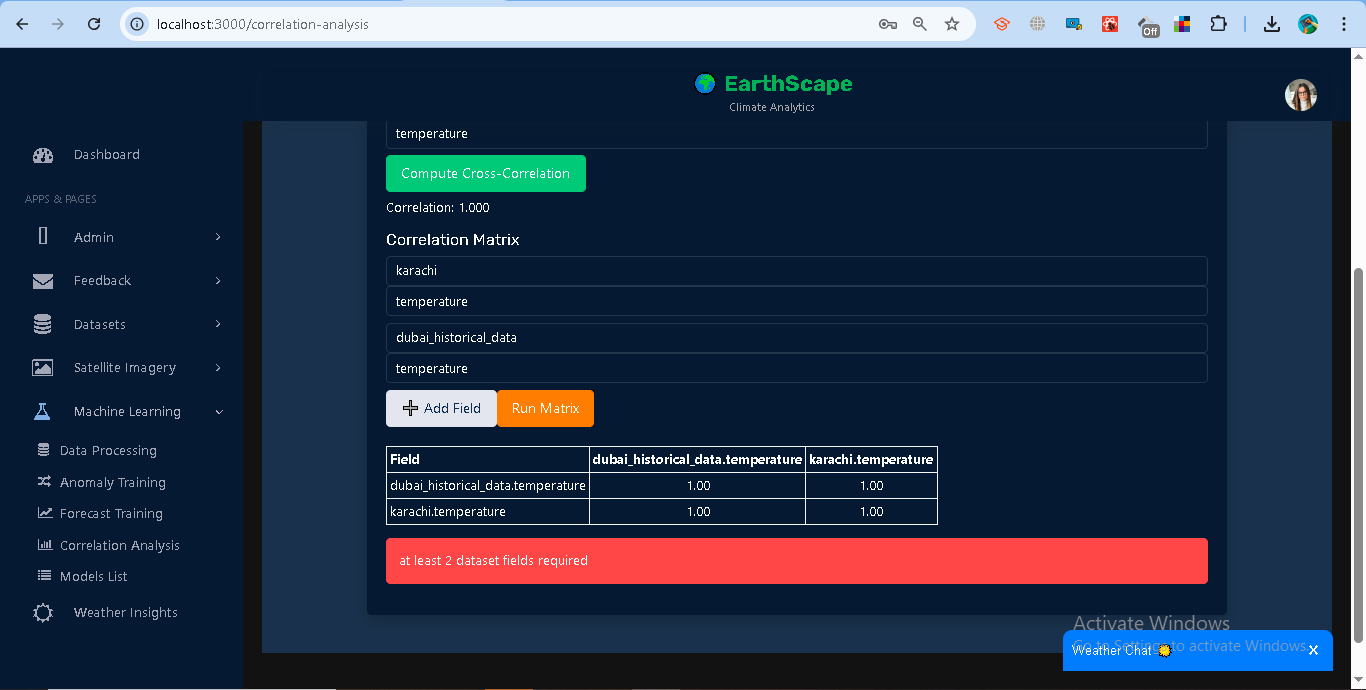
**Cross-correlation**

* Compares similarity between two dataset fields.
* Steps:  
  1. Select **Dataset 1 + Field**.
  2. Select **Dataset 2 + Field**.
  3. Compute correlation.
* Results: Single correlation score (between -1 and 1).



**Correlation Matrix**

* Shows correlation between multiple dataset fields at once.
* Steps:  
  1. Add datasets + fields (minimum 2).
  2. Run Matrix.
* Results: Table of pairwise correlation values.



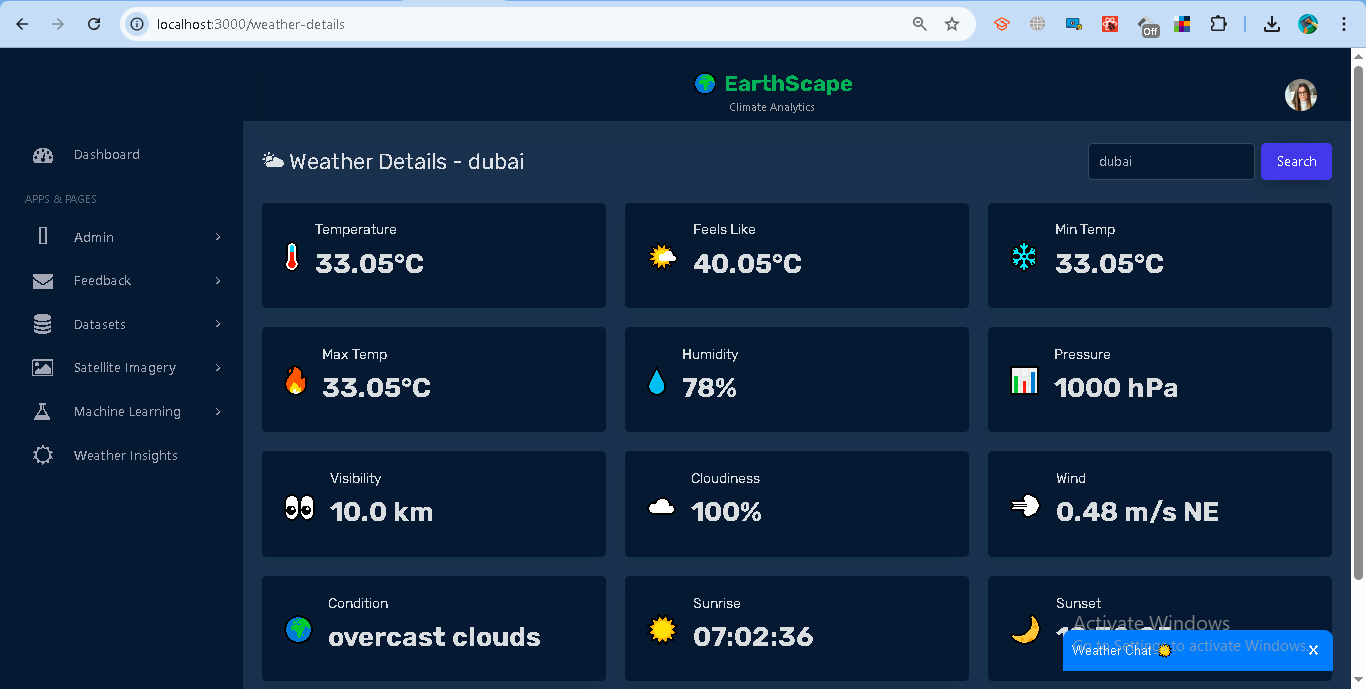
## **1.10 Weather Details**

The Weather Details module provides **real-time weather insights** using the OpenWeatherMap API. Users can search for a city and view key weather indicators in a dashboard-style card layout.

### **Features**

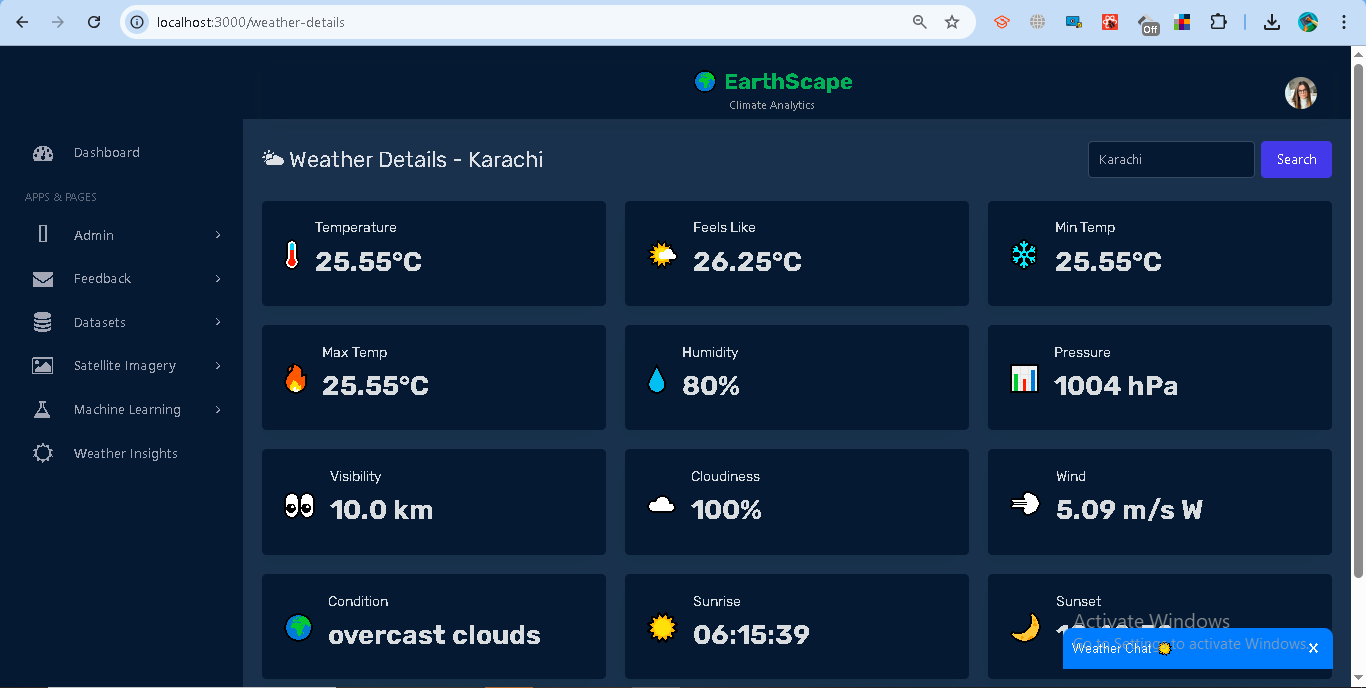
**Search and Fetch Weather**

1. Go to **Weather Insights** in the sidebar.
2. Enter a **city name** (e.g., *Dubai*) and click **Search**.
3. The system fetches live data from OpenWeatherMap.
4. The query is stored in MongoDB with a timestamp.



**Displayed Metrics**

* **Core:** Temperature, Feels Like, Min Temp, Max Temp.
* **Atmosphere:** Humidity, Pressure, Visibility, Cloudiness.
* **Wind & Condition:** Wind speed + direction, Weather condition.
* **Sun Info:** Sunrise and Sunset times (local).
* **Rain/Snow:** If available, rain/snow amount is displayed.



## **1.11 Weather ChatBot**

The Weather ChatBot provides an interactive way for users to query weather conditions for any city or country. It acts as a conversational assistant, fetching live data from the backend weather API.

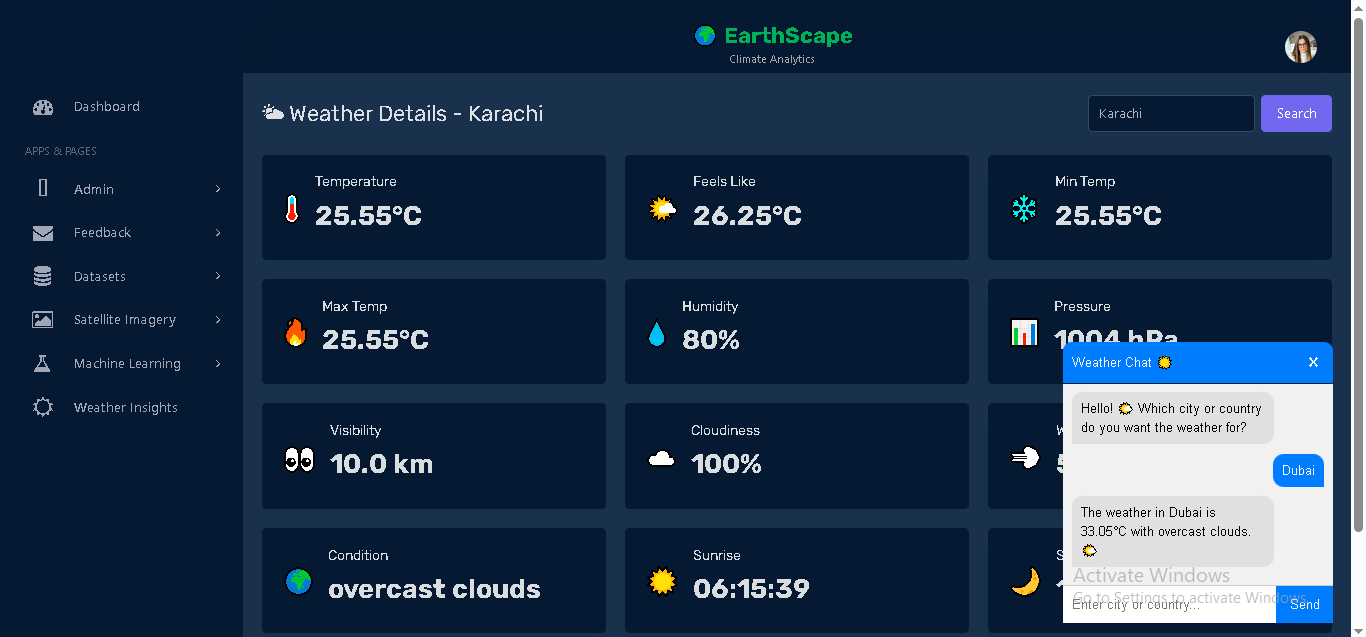
### **Features**

**Interactive Messaging**

* The bot greets the user and prompts for a city or country.
* Users type queries into the chat input.
* The bot responds with weather conditions such as **temperature** and **description**.

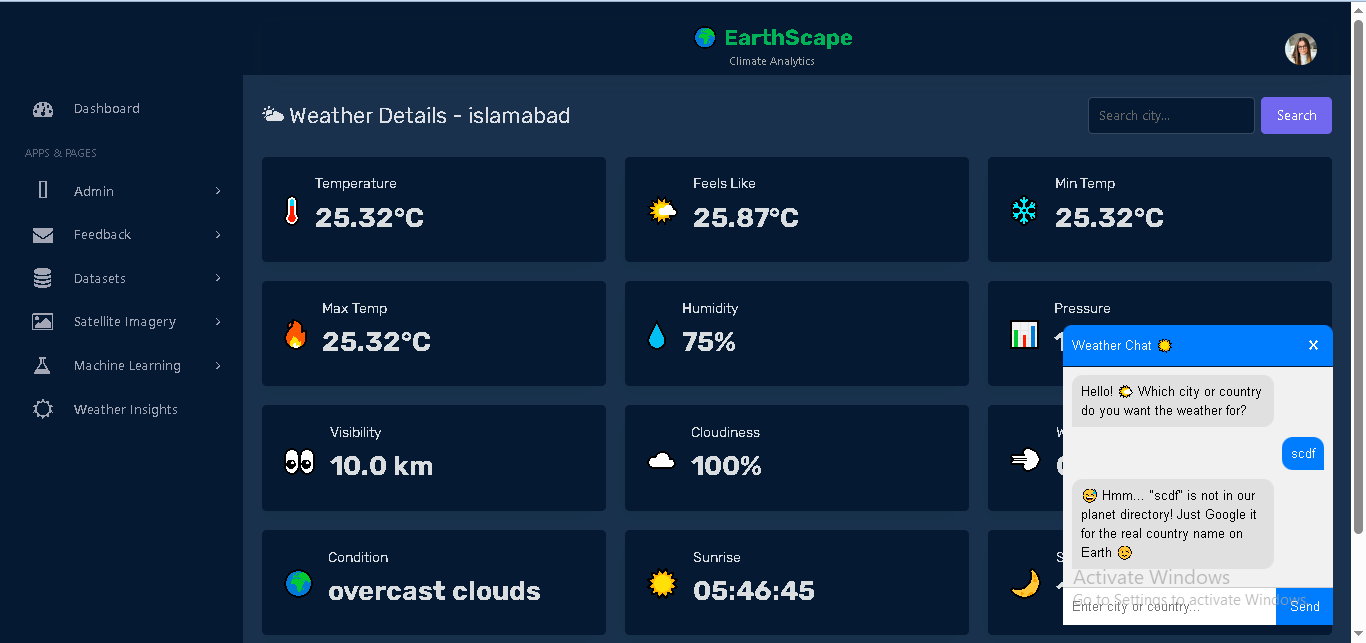
**Weather Query**

* Input: City or country name (e.g., *Dubai*, *USA*).
* Output: A conversational response, e.g.:  
   *“The weather in Dubai is 33°C with clear sky 🌤️”*

**

**Error Handling**

* If an invalid city/country is entered, the bot replies with a playful error message:  
   *“😅 Hmm... ‘X’ is not in our planet directory! Just Google it for the real country name on Earth 😉”*

**

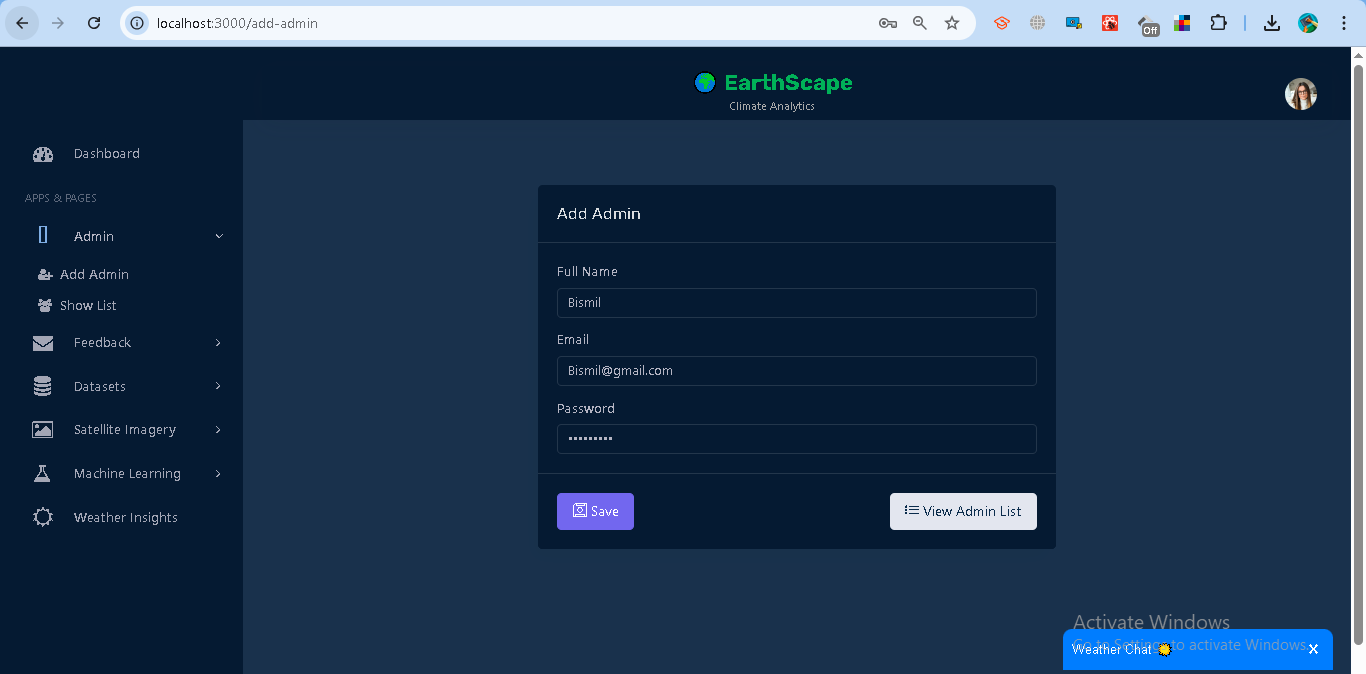
## **1.12 Admin Management**

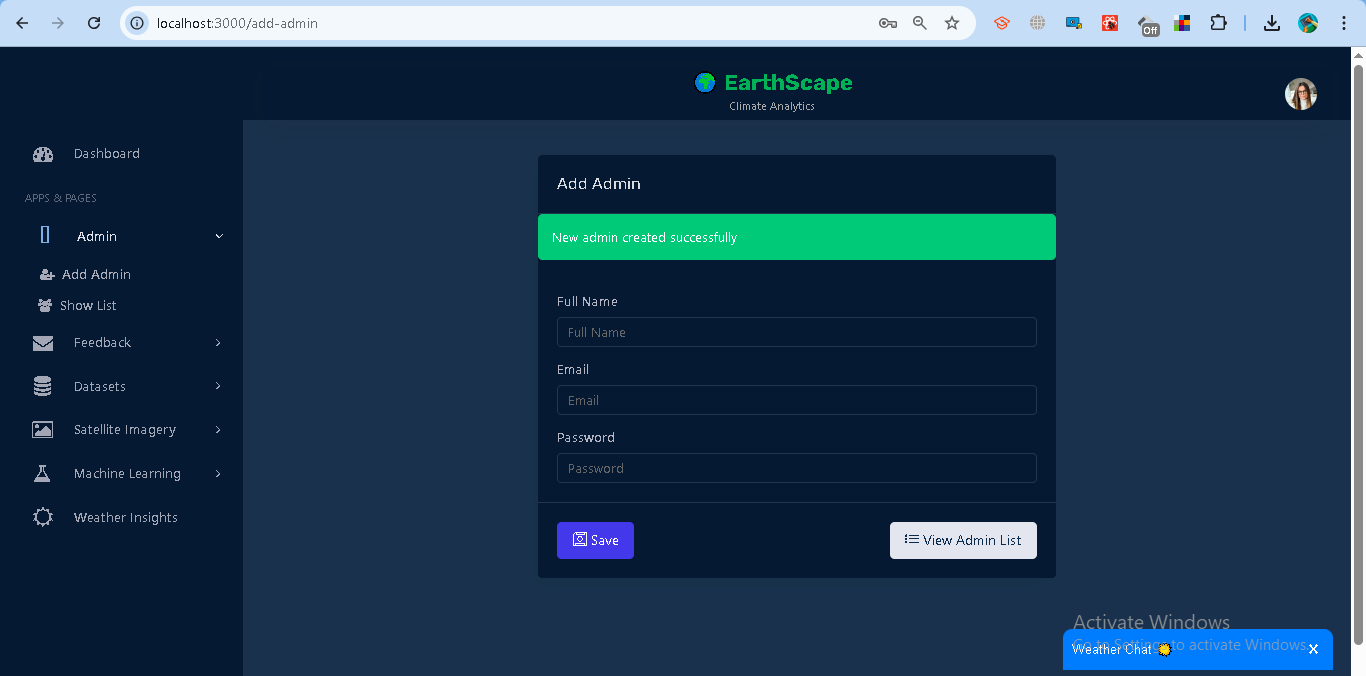
The Admin Management module allows only users with the **admin role** to manage administrator accounts.

### **Features**

**Add Admin**

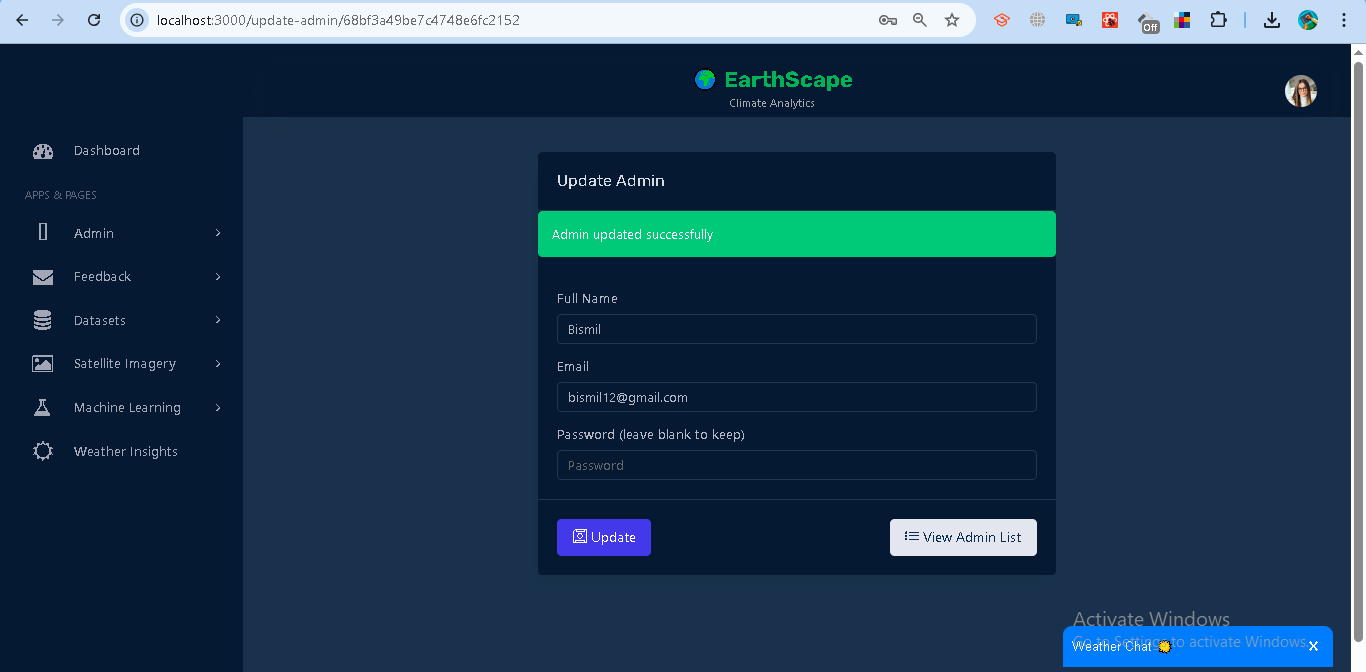
* Accessible only to admin users.
* Enter **Full Name, Email, Password**.
* On success, the new admin is stored in the system and a confirmation is displayed.





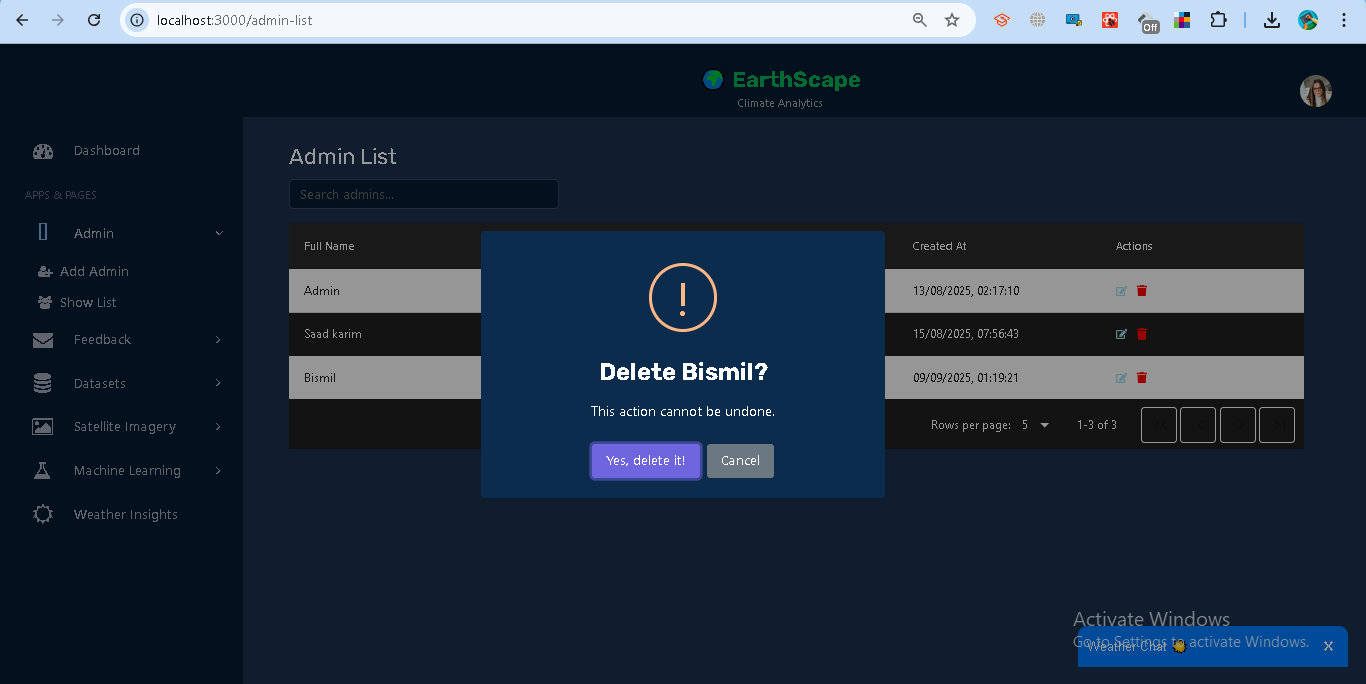
**Update Admin**

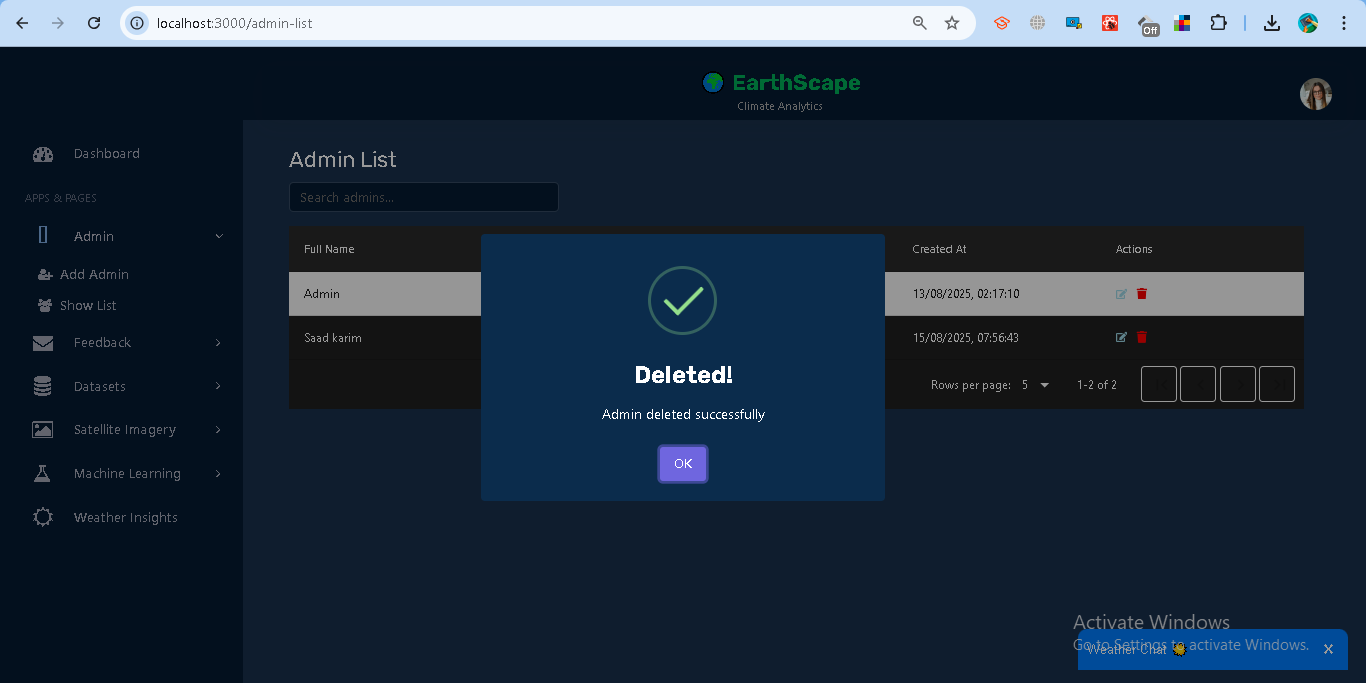
* Admins can update existing admin details such as **Full Name, Email, Password**.
* Requires selecting an admin from the list and saving changes.



**Delete Admin**

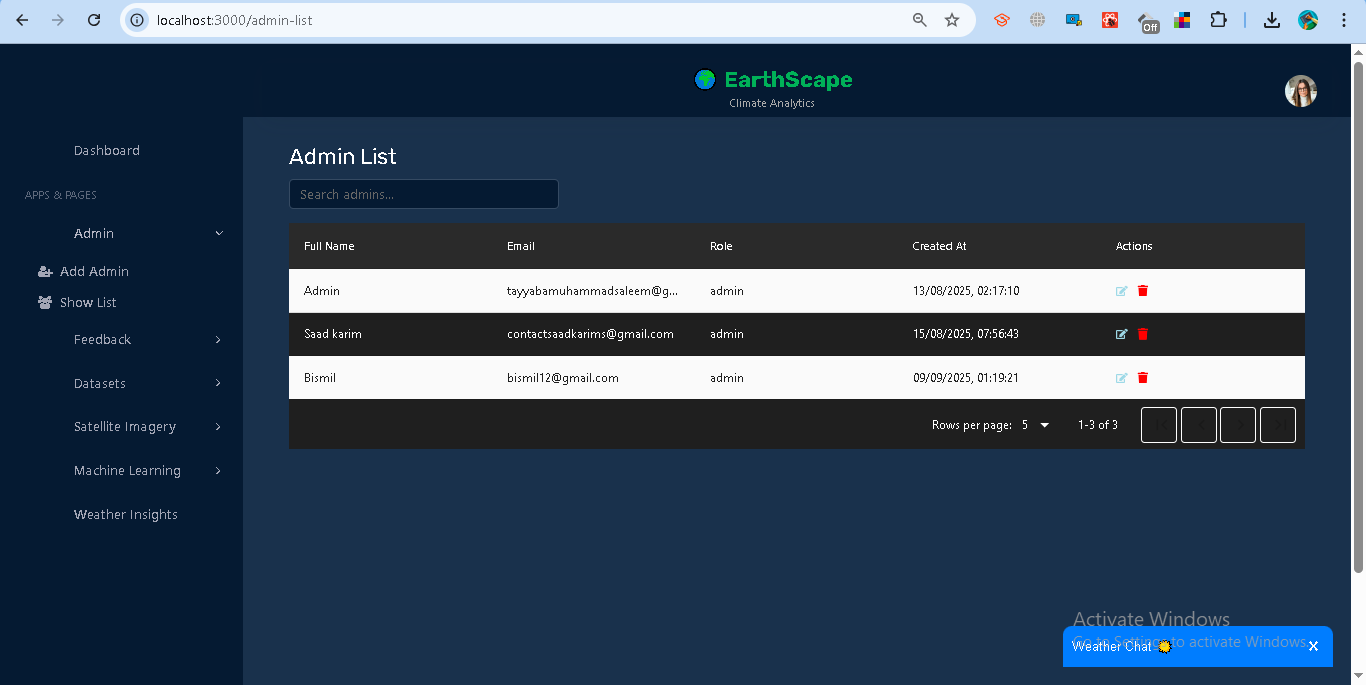
* Admins can remove an existing admin from the system.
* Requires confirmation before deletion.





**View Admin List**

* Displays all registered admin users.
* Provides options to update or delete each admin.



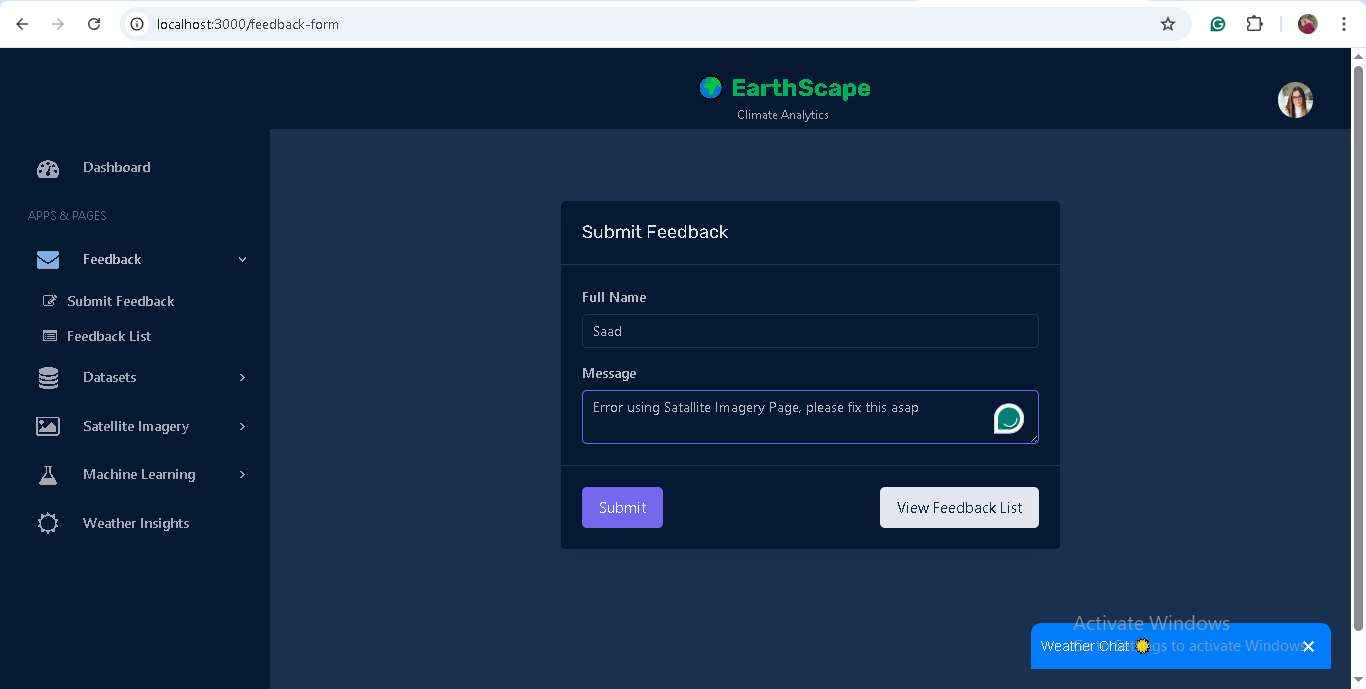
## **1.13 Feedback Management**

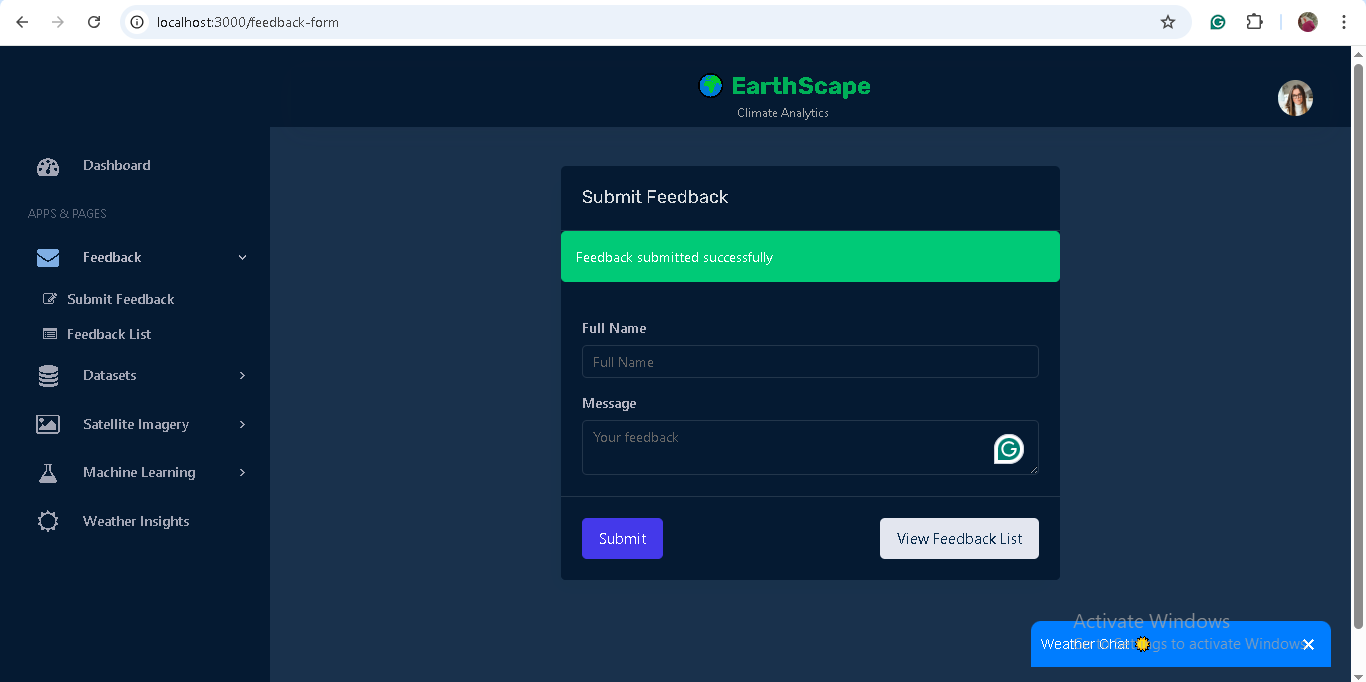
The Feedback module allows users to submit feedback and track its status, while administrators can manage and resolve feedback entries.

### **Features**

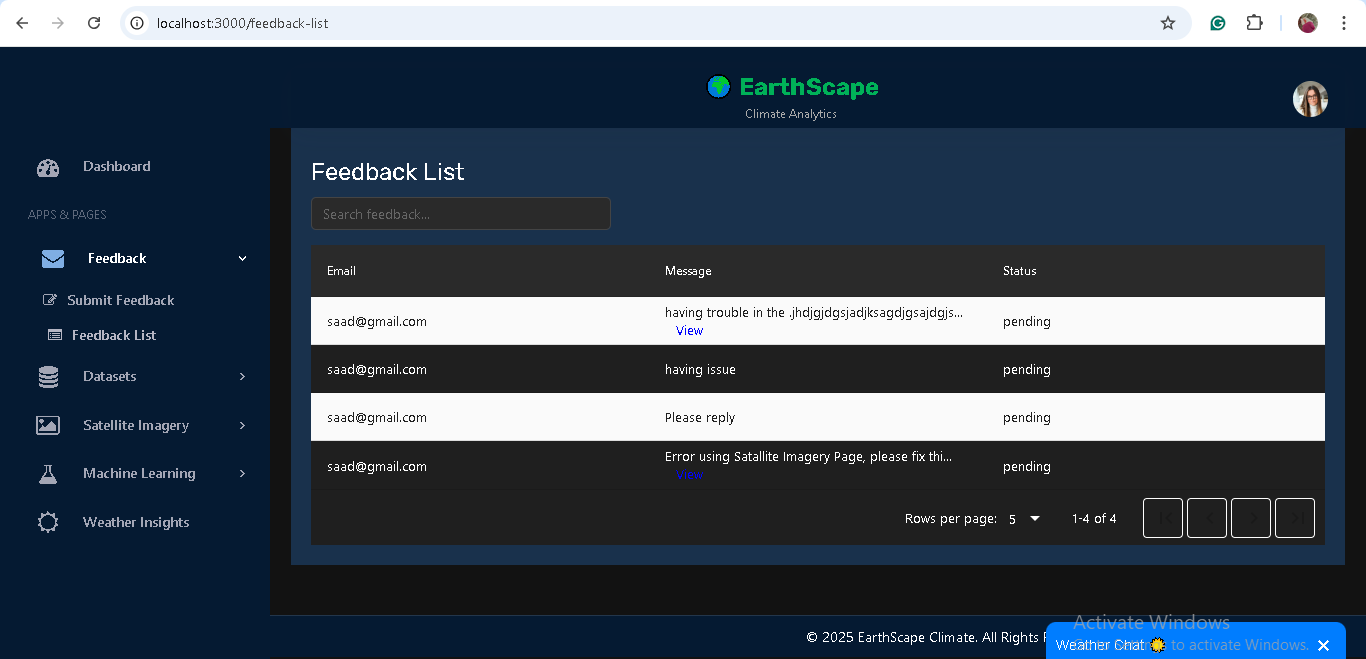
**For Analysts (Normal Users):**

* **Submit Feedback**
  + Enter **Full Name** and your **Message**.
  + The system automatically attaches your logged-in email.
  + On success, you see: *“Feedback submitted successfully.”*

**

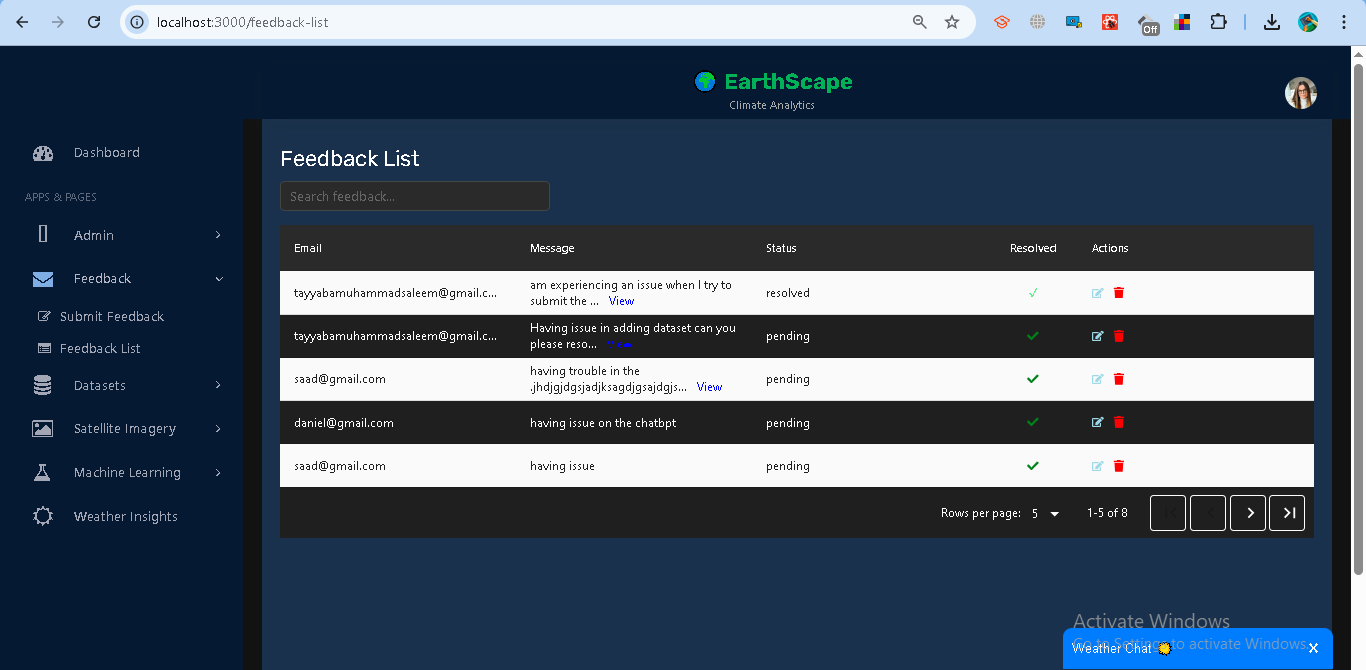
**

* **View Feedback List**
  + Shows all feedback you have submitted.
  + Displays **message, status, and date submitted**.
  + Status may be *Pending, In-Progress,* or *Resolved*.

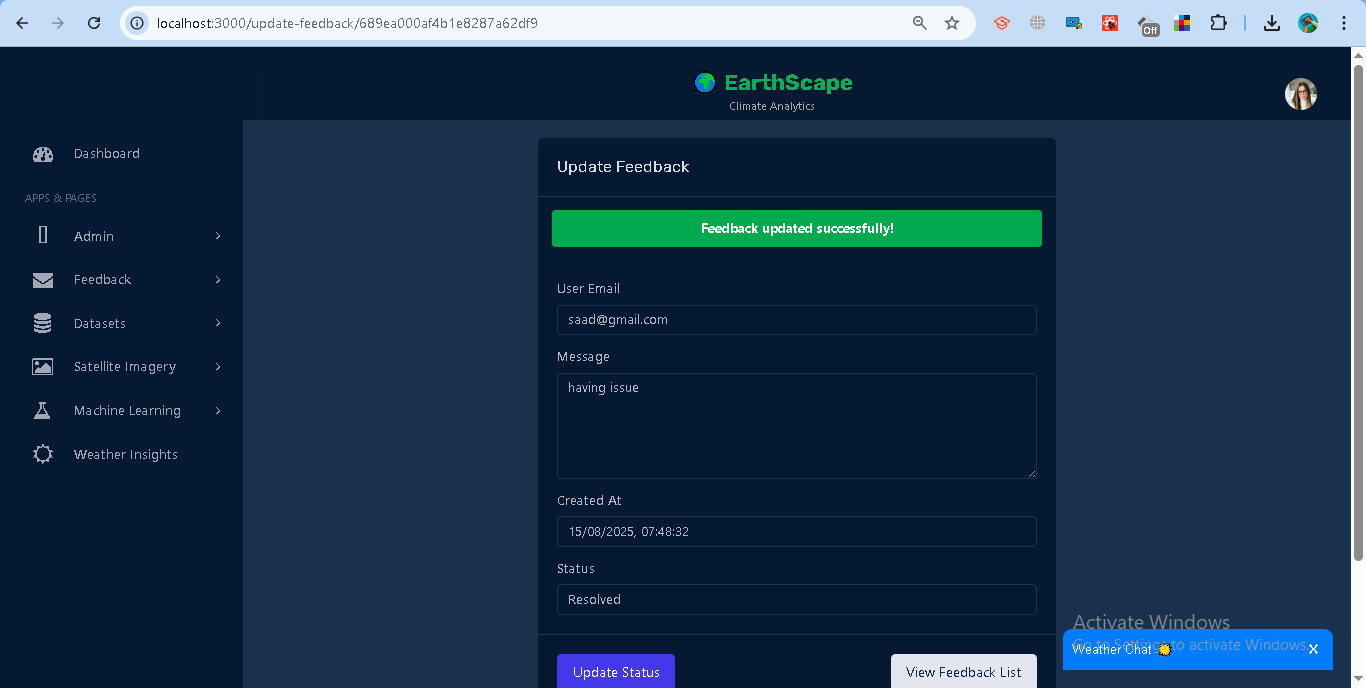


**For Administrators (Admin Role):**

* **View All Feedback**
  + Access to all feedback submitted by any user.
  + Useful for monitoring system-wide feedback.



* **Update Feedback Status**
  + Change status of feedback to *Pending, In-Progress,* or *Resolved*.
  + Helps track progress of reported issues or suggestions.



* **Delete Feedback**
  + Remove feedback entries when necessary.
  + Requires confirmation to prevent accidental deletions.

