# Task 4: Application Development

The last component involved creating an easy-to-use, multipage GUI to perform any operation on the dataset and the prediction model. The developed application was created with Python programming language and has been implemented in the Streamlit environment since the latter enables developers to create prototypes of data applications with good GUI feedback.

#### **1. Application Structure**

The application consists of a **multi-page layout** with the following sections, accessible via a sidebar navigation menu:

1. **Data Overview** This section provides a summary of the dataset, including:
   * Number of rows and columns
   * Column names and data types
   * Sample data (via df.head())
   * Missing values overview
   * Data distribution summary
2. **Exploratory Data Analysis (EDA)** The EDA section includes:
   * Interactive visualizations using **Plotly** and **Matplotlib**, such as:
     + Line charts of PM2.5 levels over time
     + Bar charts comparing average pollution levels by station
     + Heatmaps for correlation between variables
   * Dropdowns and sliders allow users to filter data by station, period, or pollutant (Huang et al., 2023).
   * Statistical summaries and charts help users understand seasonal and spatial trends in air quality.



**Figure: Application Development**

(Source: Google-colab)

1. **Modelling and Prediction** This page allows users to:
   * Input meteorological and pollutant variables through sliders and number input boxes.
   * Select a station and time of day to contextualize the prediction.
   * View predicted **PM2.5 levels** using the trained **XGBoost model**.
   * Display model metrics (MAE, RMSE, R²) from the testing phase for transparency.

#### **2. Technical Implementation**

* The model was saved using **joblib**, allowing it to be loaded into the app without retraining.
* Inputs from the prediction form are preprocessed (feature scaling and encoding) to match the model’s training schema.
* The GUI dynamically updates visualizations based on user interactions, enhancing the exploratory experience.

**3. User Experience and Accessibility**

This approach also anticipated that the developed application should not require any technical knowledge to be used. To make the work of the user as easy as possible, there are clear labels, tooltips, and section headings that help to navigate through the analysis and prediction phase. For flexibility between sections, Modularity concepts are applied, and each section consists of a CSS presentation and has collapsible features to manage its contents.

They have succeeded in developing an application that brings out the complexity of noraq into simple details and facts that can easily be understood by the people involved. It involves real-time data used for decision making that can be valuable to the policymakers, the environmental gurus, and the general public who like to know the level of pollution in Beijing.

# 6. Task 5: Version Control

Revision control is an essential tool used in the development of computer programs that allows changes to the code to be managed, development to be done in cooperation, as well as retaining a record of all changes made to the code. In the implementation of this project, Git was chosen as the versioning system to be used, while GitHub acted as the host for the remote repository for the project.

**1. Repository Setup and Branching**

In the beginning, the structure of the project was created, and the GitHub repository was created and connected to the local environment (Lu et al., 2020). The main branch was a master copy – the code in it was constantly working in production; other branches were created for each major job:

* data-handling
* eda-analysis
* model-building
* app-development
* documentation

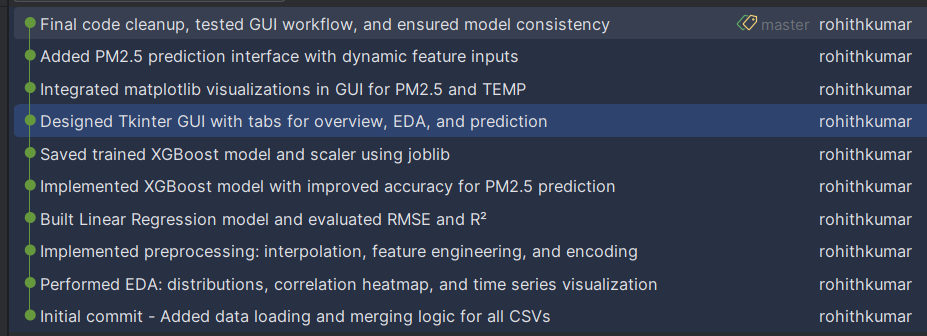
This branching strategy ensured that each component of the project could be developed and tested independently without interfering with the main branch.

**2. Commit Practices**

During the whole process of developing, clear and concise but descriptive commit messages were used to make the changes easily understandable. For instance:

* “Loaded and merged air quality datasets from selected stations”
* “Performed missing value treatment and feature engineering”
* “Built XGBoost model and evaluated performance”
* “Implemented Streamlit multi-page app structure”

Each commit represented a specific, atomic change, promoting traceability and facilitating debugging when necessary.



**Figure: 10-Git Commit Log**

(Source: Google-colab)

**3. Collaboration and Backup**

Even though this project is individual, using GitHub as a platform has its backup system and versions earlier than others. But if the experience allows it, in the future, more elaborate scenarios, based on pull requests and code reviews, are possible in a team environment.

**4. Screenshot Evidence**

A screenshot of the GitHub repository containing information on the commit and branch activity, as well as the files in the project, was taken and included to accomplish this task (Nantasenamat et al., 2023). It befits the Learning Outcome 5 (LO5) of managing version control systems and collaborative tools, which requires students to utilize version control tools in actual projects.

The application of version control has brought positive changes on the organization of the projects and created a basis of better model of teamwork in realistic software development settings, as well as improved traceability and quality of the project work.