Diagram of runtime components

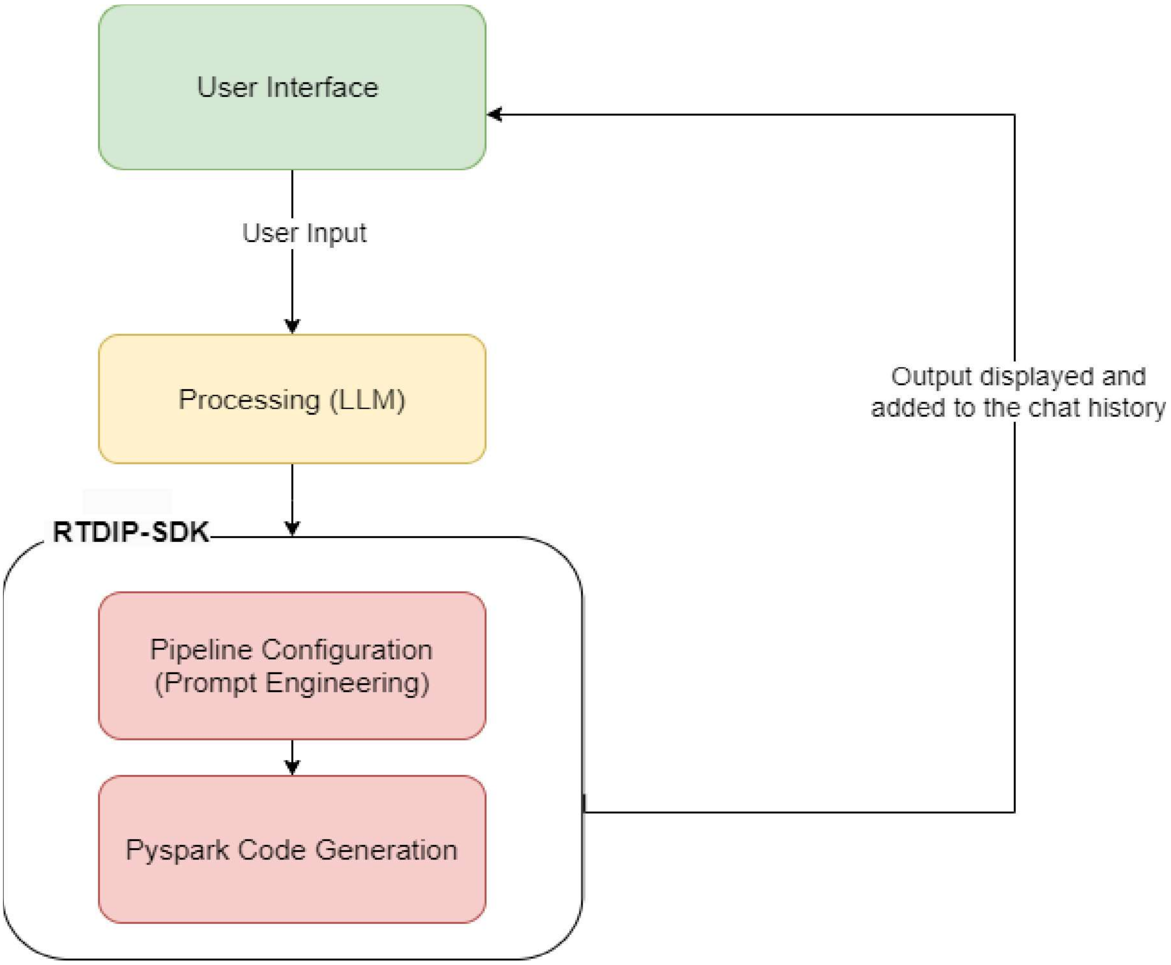
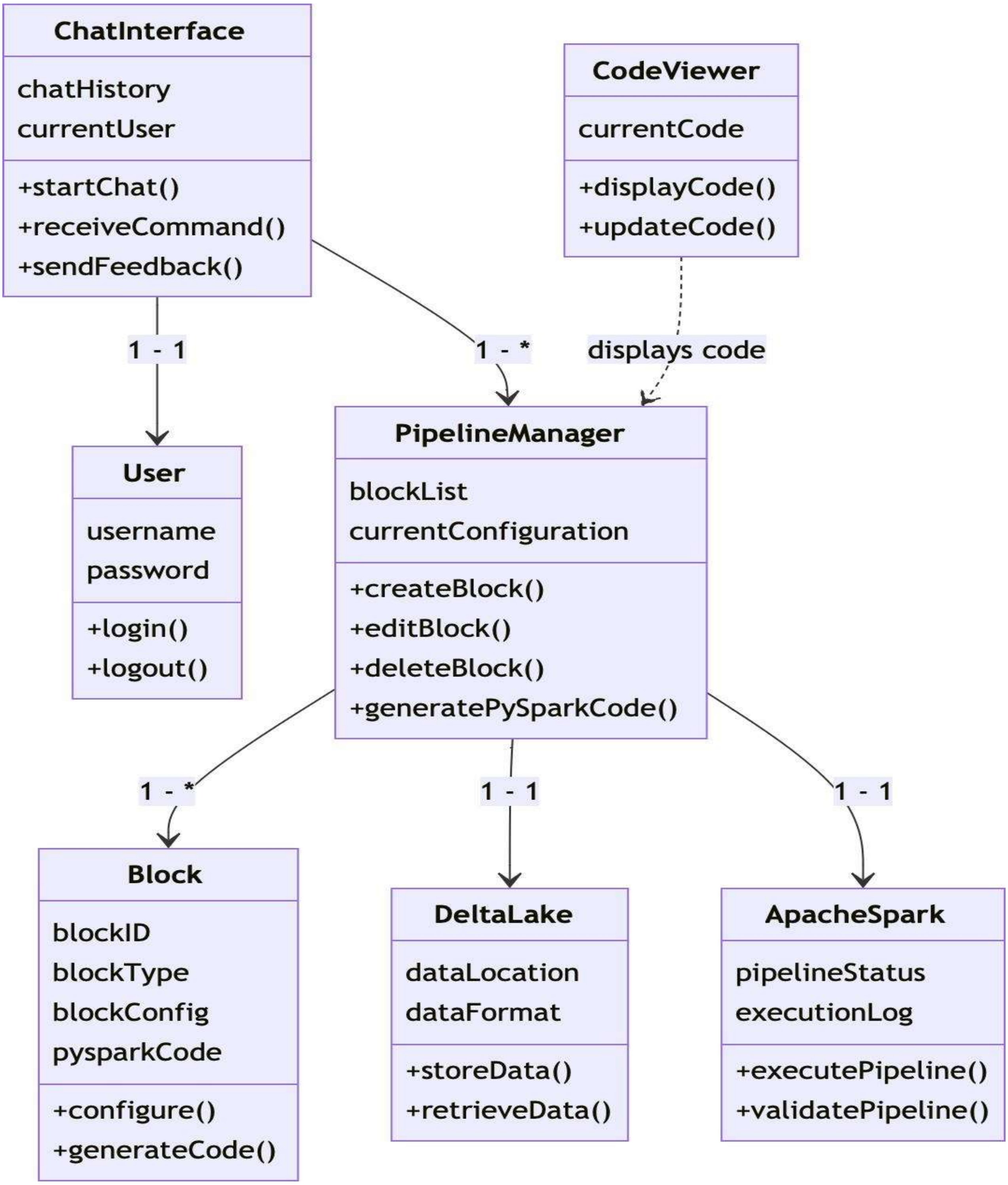

User Interface
User Input
Processing (LLM)
RTDIP-SDK
Pipeline Configuration (Prompt Engineering)
Pyspark Code Generation
Output displayed and added to the chat history

Diagram of runtime components



Technology Stack Summary

The project highlights the development of a chat AI-based user interface, enabling users to interact with the system. Through this interface, users can create and configure RTDIP data pipelines. This configuration includes defining block configurations, representing the steps in the data pipeline, and writing PySpark code for specific data processing tasks.

Python: is a widely-used, high-level programming language known for its simplicity and readability. It is used as the programming language for implementing various components of our project. It is likely used for developing the user interface, implementing the chat AI, and for writing PySpark code for data processing.

RTDIP-SDK : is the core technology around which our project is built. RTDIP stands for Real-Time Data Integration Platform. It is a software development kit (SDK) that is used to configure pipelines. It is common in data processing, ETL (Extract from different sources, Transform and Load using Data Lake).

Apache Spark: is a distributed data processing framework that plays a crucial role in the data pipeline. It is used for performing data transformations, data cleaning, and various data processing tasks. PySpark, which is the Python API for Apache Spark, is mentioned as a key component.

Hugging Face Transformers: is a Python library for natural language processing (NLP) that provides access to a vast selection of pre-trained NLP models and tools.

Delta Lake: is used as the storage layer, serving as the sink of the ETL process.

In summary, the technology stack for this project is comprehensive and dynamic, integrating a range of tools and frameworks to facilitate data integration, processing, and the development of an AI-driven user interface.

Explanation of the diagrams and choices for the ChatAI

Modular Architecture Diagram : The modular architecture diagram illustrates the separation of concerns among different modules such as Input Processing, Configuration Generation, Execution, SDKIntegration, UI, and Monitoring & Logging. This separation promotes maintainability, scalability, and ease of testing.
Data Flow Diagram : The Data Flow diagram showcases how data traverses through the system, from userinput to configuration generation, and finally to execution within the Apache Spark cluster integrated with RTDIP-SDK. It helps in understanding the flow of data and interactions among different components.

Architectural Choices Explanation

1. Utilizing Apache Spark:

- Real-time Processing: Apache Spark's ability to handle real-time data processingis crucial for the project requirements.
- Scalability: Spark's distributed computing capabilities ensure that the system canscale with the data volume.

2. Integration with RTDIP-SDK:

- Specialized Processing: RTDIP-SDK provides specialized libraries for real-timedata and image processing which are essential for the project.
- Ease of Integration: The SDK's compatibility with Apache Spark allows for aseamless integration, reducing development time and potential integration issues.

3. Employing a Language Model for Configuration Generation LLM (Maybe MPT):

- Intuitive Interaction: The LLM enables an intuitive interaction for users tospecify their requirements, making the system user-friendly.
- Automation: Automating the generation of pipeline configurations significantlyreduces the manual effort and minimizes the scope of errors.

4. Web-based User Interface:

- Accessibility: A web-based UI ensures accessibility from various devices andplatforms.
- Real-time Feedback: Provides real-time feedback to users regarding thegenerated configurations and any runtime issues, enhancing user experience.