PROJECT / RELEASE

Project Design Document

G2

Kiara Toska <kt7988@rit.edu>
Mohamed Amgad <ma3568@rit.edu>
David Kraljic <dk9612@rit.edu>
Feridun Emre Tuncer <fet3523@rit.edu>
Emanuel Ivan Mlikota <em5561@rit.edu>

Project Summary

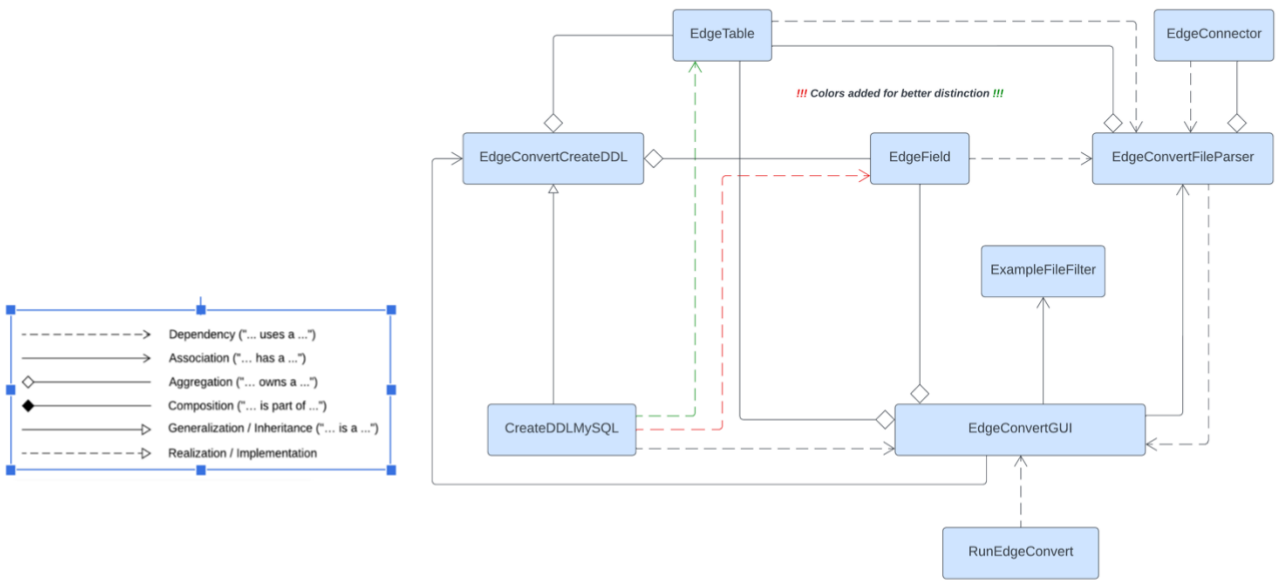
This section provides a brief overview of the project. Do not cut and paste the project's assignment exactly as is. You may use sections of the assignments, but summarize the project in your own words; a few paragraphs are sufficient.

Design Overview

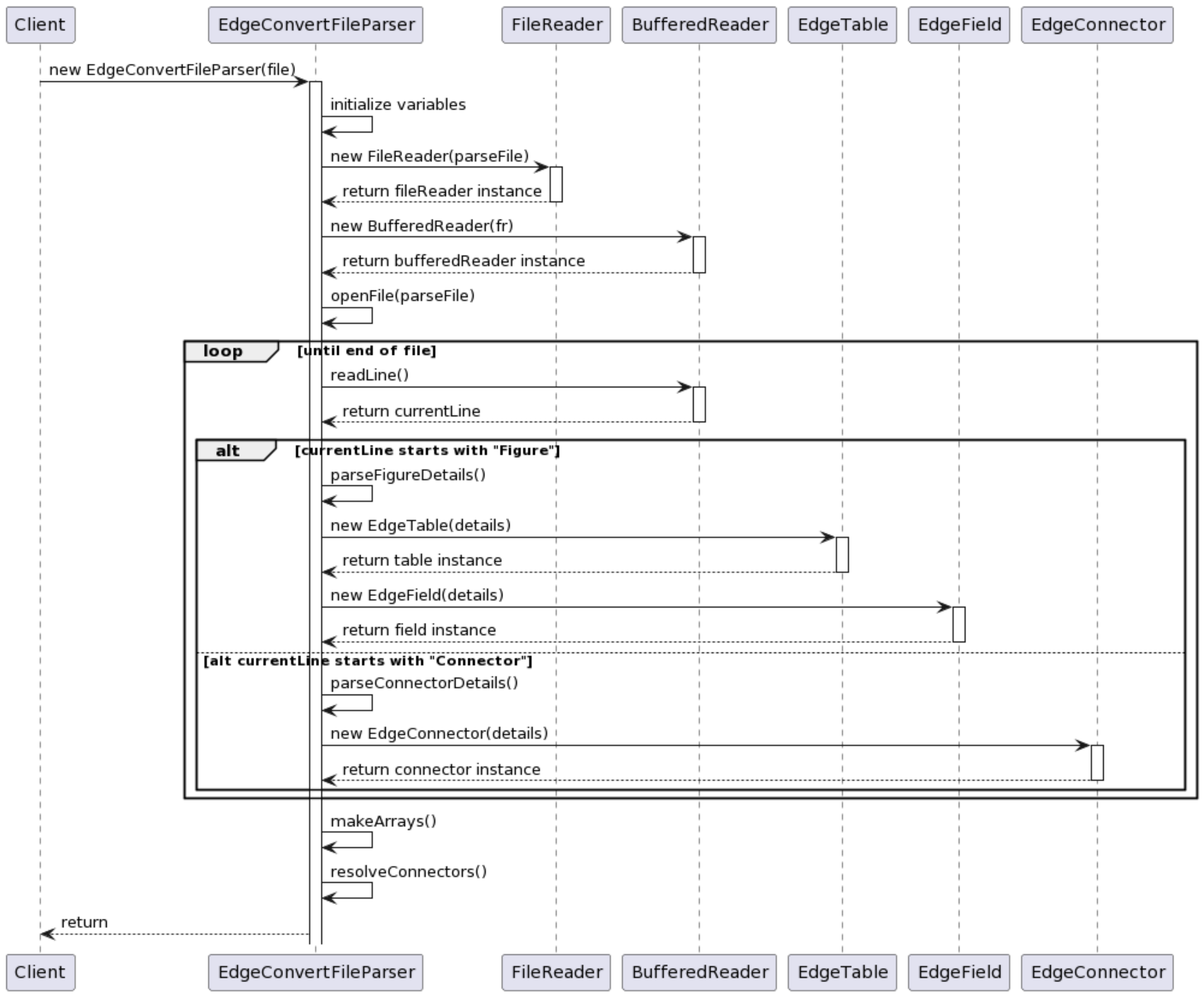
The design overview is the narrative that captures the thought process and evolution of the design from the preliminary design sketch through final implementation. The narrative should support how the project design addresses good design principles: separation of concerns, high cohesion, low coupling, support for extendibility, etc.

It is equally important to document design decisions that did not go as anticipated, as it is decisions that worked out well. This is extremely helpful background for future readers of the document to help them avoid solution paths that already had been attempted when extending the project with new or modified features. It is common for some design documents to have an entire section dedicated to “rejected alternatives”.

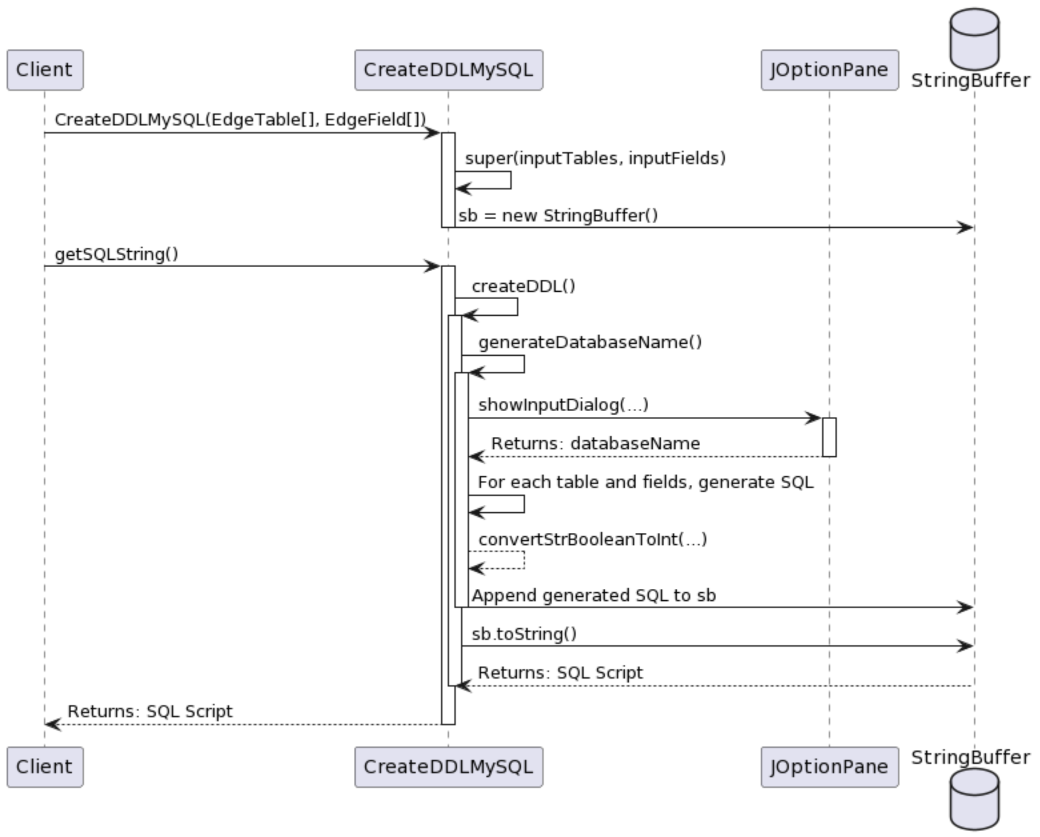
Class Diagram #1


EdgeTable
EdgeConnector
!!! Colors added for better distinction !!!
EdgeConvertCreateDDL
EdgeField
EdgeConvertFileParser
ExampleFileFilter
CreateDDLMySQL
EdgeConvertGUI
RunEdgeConvert
Dependency ("... uses a ...")
Association ("... has a ...")
Aggregation ("... owns a ...")
Composition ("... is part of ...")
Generalization / Inheritance ("... is a ...")
Realization / Implementation

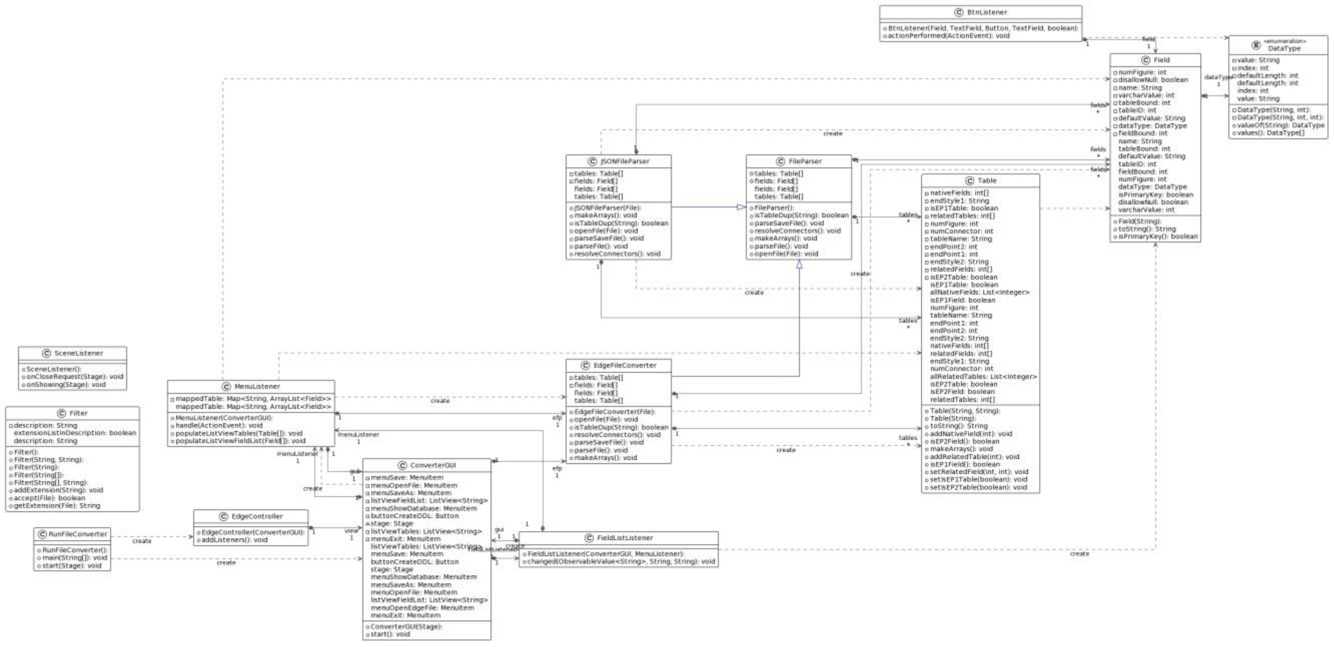
Sequence Diagram #1



Sequence Diagram #2



Overall System Structure



Subsystems

The provided UML diagram portrays several classes, enumerations, and their interrelationships which can be grouped into various subsystems based on their functionality and the responsibility they carry within the application. In this description, we will segment them into subsystems, illustrate their internal relations, and provide a brief overview of each.

File Conversion Subsystem

[EdgeFileConverter, JSONFileParser, FileParser, RunFileConverter, and related classes]

This subsystem is primarily responsible for the conversion of files. It showcases the use of the Strategy Pattern, where various file parsing strategies (EdgeFileConverter, JSONFileParser) can be used interchangeably.

EdgeFileConverter: Handles the conversion of files specifically of type 'Edge'. This class has compositions with Table and Field classes, implying it creates and manages instances of these classes.

JSONFileParser: Similar to EdgeFileConverter but tailored for parsing JSON files. It inherits from FileParser, hinting at a possibility of shared functionalities between different parsers.

FileParser: Serves as a generic file parsing class, which other specific file parsers like JSONFileParser can extend.

RunFileConverter: Acts as an entry point, creating instances of ConverterGUI and EdgeController.

Collaborates with: [GUI Subsystem]

GUI Subsystem

[ConverterGUI, MenuListener, FieldListListener, BtnListener, and related classes]

This subsystem manages the user interface and user interactions of the application.

ConverterGUI: Central GUI class, having compositions with EdgeFileConverter, FieldListListener, and MenuListener. This class is pivotal in updating and reflecting data changes on the interface.

MenuListener: Listens to menu-related events within the GUI. It has associations with ConverterGUI and EdgeFileConverter.

FieldListListener: Listens to changes or interactions related to a field list in the GUI. It interacts directly with ConverterGUI and MenuListener.

BtnListener: Handles button-related events and has a relation with the Field class, implying it may perform actions based on certain field properties.

Collaborates with: [File Conversion Subsystem]

Data Modeling Subsystem

[Table, Field, DataType, and related classes]

This subsystem deals with the data modeling aspect of the application.

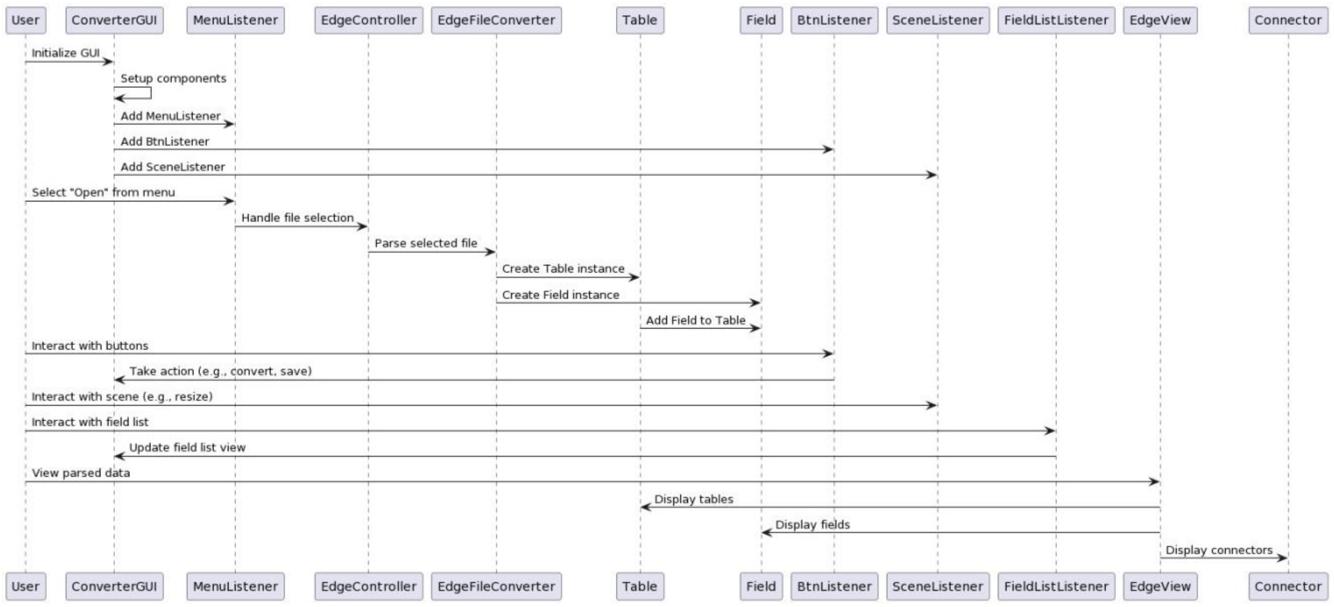
Table: Represents the structure and attributes of a table within the application. It has associations and compositions with the Field class, showcasing the table-to-field relationships.

Field: Models the structure of fields inside tables. It has an association with DataType, indicating the field can have different data types.

DataType: An enumeration that defines various possible data types a Field can have.

Collaborates with: [File Conversion Subsystem], [GUI Subsystem]

Sequence Diagrams



The given sequence diagram represents a series of interactions among several components within a graphical user interface (GUI) application, particularly one that involves file parsing, data conversion, and data visualization. The participants (or entities) involved in the process are: User, ConverterGUI, MenuListener, EdgeController, EdgeFileConverter, Table, Field, BtnListener, SceneListener, FieldListListener, EdgeView, and Connector.

Initialization and Setup

User initiates the process by triggering the ConverterGUI (gui) to initialize the graphical user interface.

ConverterGUI subsequently performs self-activities to set up its various components.

ConverterGUI then establishes communication with MenuListener, BtnListener, and SceneListener by adding them, implying that it registers these listeners to catch and respond to respective events in the future.

File Selection and Parsing

User selects the "Open" option from the menu, notifying the MenuListener.

MenuListener handles this interaction and communicates with EdgeController to manage file selection.

EdgeController engages with EdgeFileConverter to parse the selected file.

EdgeFileConverter, upon parsing the file, generates instances of Table and Field, which respectively embody structured data and attributes extracted from the file.

Table then integrates the Field instance into its structure.

User Interactions and Data Manipulation

User interacts with buttons in the GUI, prompting BtnListener to take appropriate actions, which might include data conversion, saving, or other functionalities, interacting back with ConverterGUI to perform the desired actions.

User adjusts or interacts with the visual scene (possibly changing the size or view), and SceneListener captures this event.

Furthermore, the User engages with a field list, and the FieldListListener responds to these interactions, updating the GUI accordingly to reflect any changes or selections made in the field list.

Data Visualization

User accesses EdgeView to visualize the parsed data.

EdgeView then interfaces with Table and Field to retrieve and display the table and field data visually.

Additionally, EdgeView communicates with Connector, suggesting that it might visualize connections or relationships between different data points or entities. The exact nature of "connectors" isn't fully detailed in the sequence, but it implies a role in graphically linking or demonstrating relations between elements within EdgeView.

Summary

This sequence diagram portrays a user-driven data handling and visualization application. Beginning from initializing a GUI to parsing data from a selected file, interacting with various GUI components, and ultimately visualizing the parsed data, the user navigates through a series of steps facilitated by dedicated listeners and controllers. This results in an interactive experience where the user can open, manipulate, and visually explore data, with various system components working in tandem to respond to user inputs and ensure accurate, coherent visual representation.