Group symbol: **3**

Team: **3**

Project title: **wrtext**

Team members (*filled by PM, Team Leader*):

No	Name	Surname	Student ID	Role
1	Gabriele	Simoni	293981	*PM, Team Leader*
2	Hüdalfa Bera	Dalgın	293988	*Team member*
3	Nozomi Malke	Shirasaki	288599	*Team member*
4	Erik	Parra Mejido	293864	*Team member*
5	Cédric Minh	Prétet	293891	*Team member*

3. Design **(S3)**

3.1. Logical Software Architecture

The logical architecture aspect of the system should be present as a Component Diagram expressed in UML 2.5.
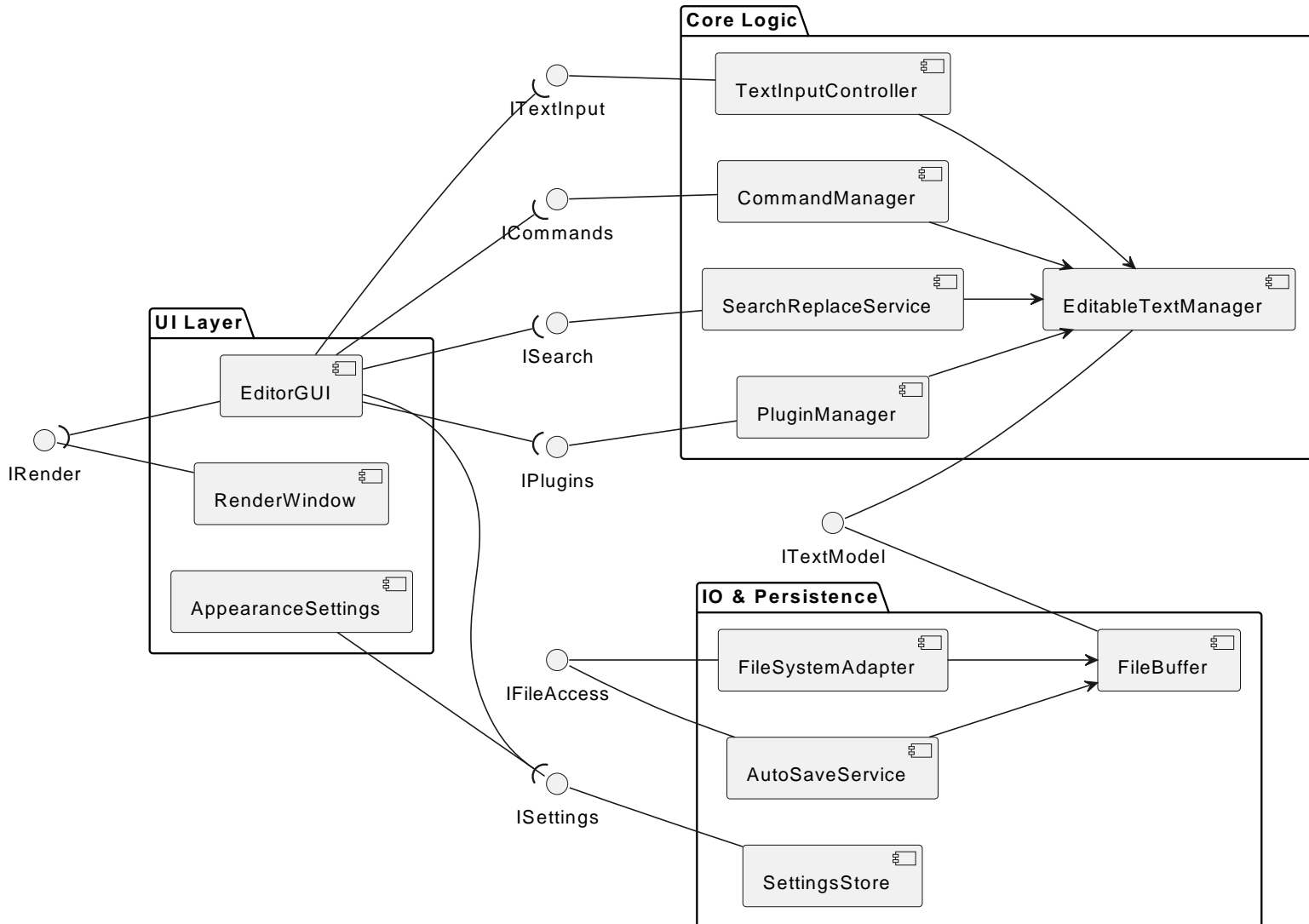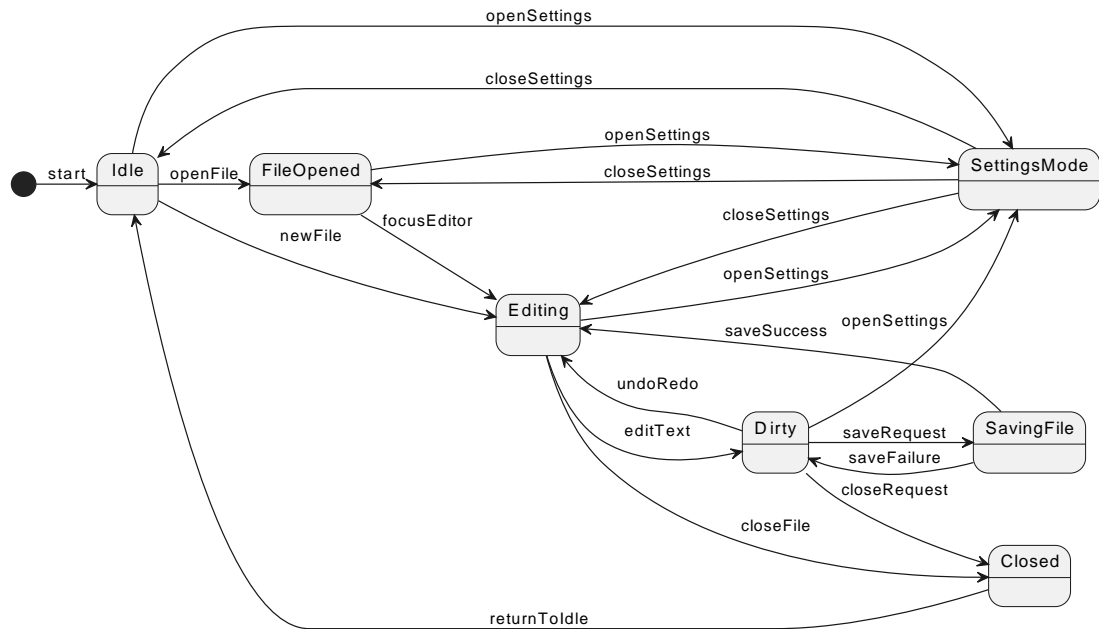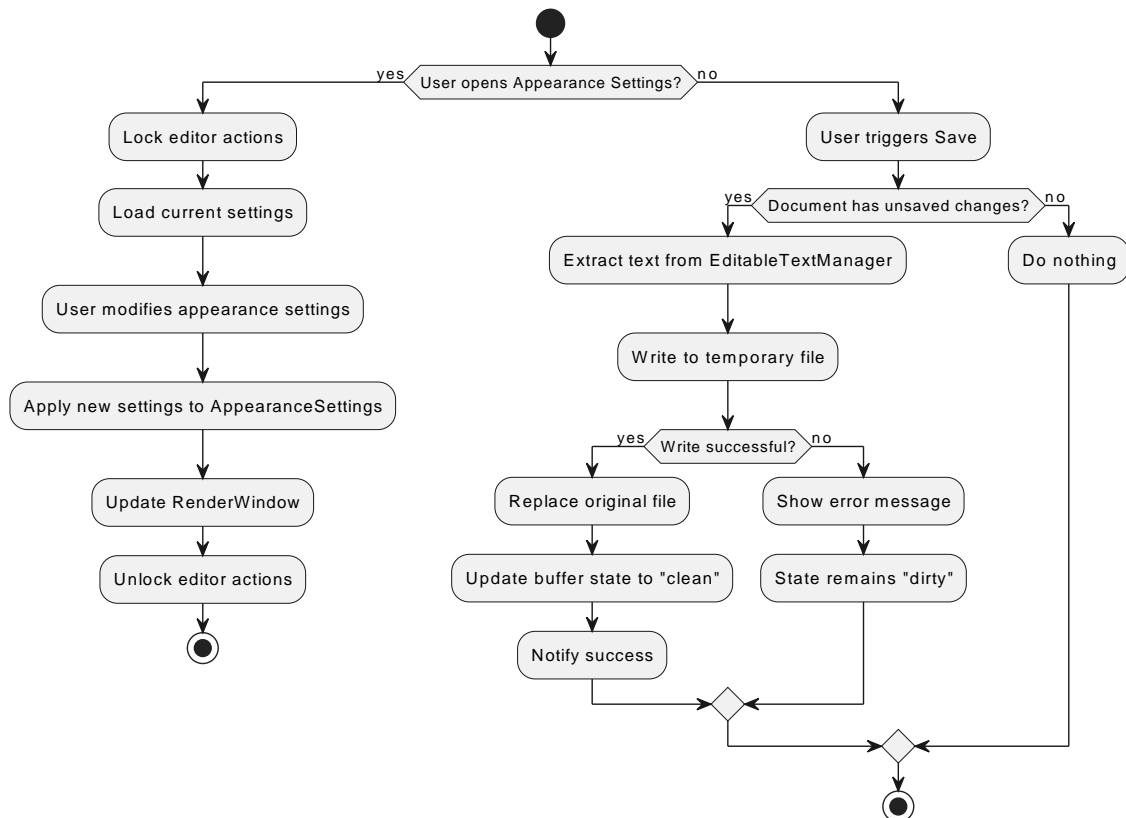
3.2. Business Logic Model

3.2.1. Behavioural Model
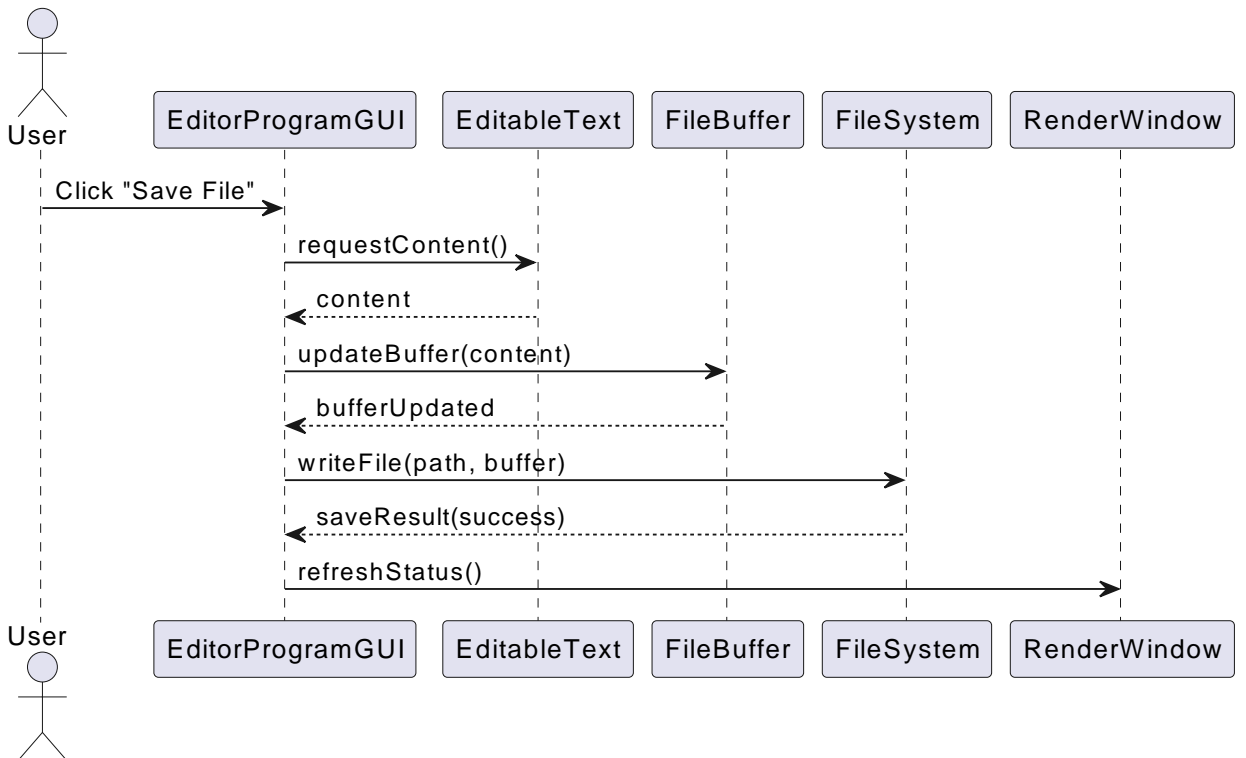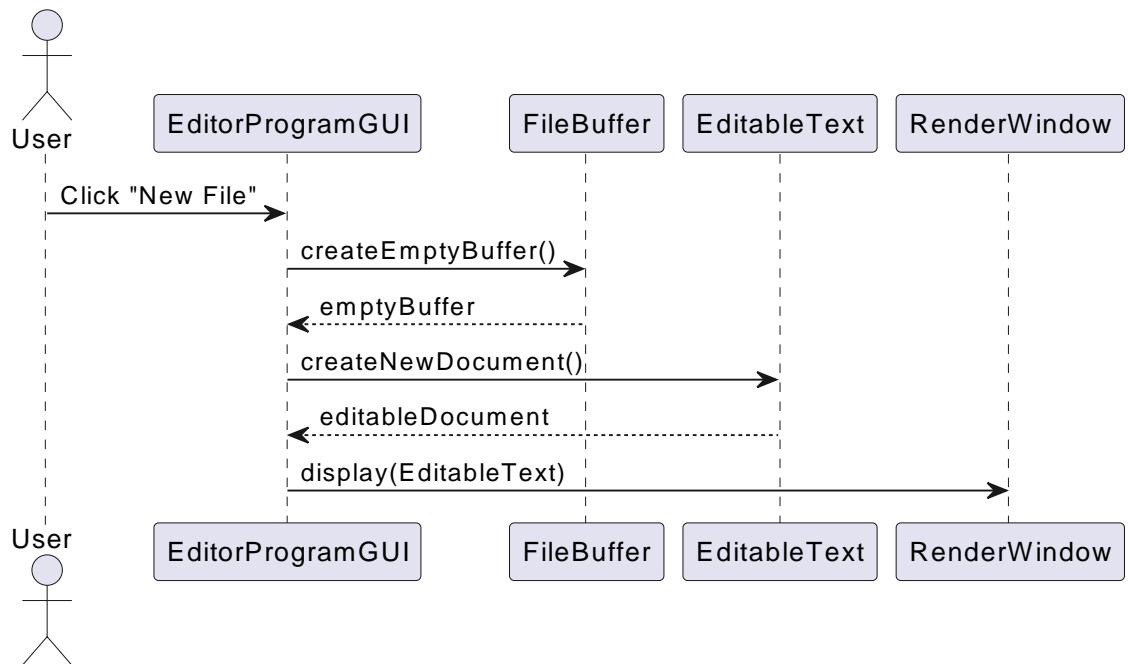
In this section present:

o   *state diagram for whole the system*

behaviour of 2 crucial use cases using the UML 2.5 diagrams, namely sequence diagram, and activity diagram.

Sequence Diagram 1: New File

- User → EditorProgramGUI: Click "New File"
- EditorProgramGUI → FileBuffer: createEmptyBuffer()
- FileBuffer --> EditorProgramGUI: emptyBuffer
- EditorProgramGUI → EditableText: createNewDocument()
- EditableText --> EditorProgramGUI: editableDocument
- EditorProgramGUI → RenderWindow: display(EditableText)

Sequence Diagram 2: Save File

- User → EditorProgramGUI: Click "Save File"
- EditorProgramGUI → EditableText: requestContent()
- EditableText --> EditorProgramGUI: content
- EditorProgramGUI → FileBuffer: updateBuffer(content)
- FileBuffer --> EditorProgramGUI: bufferUpdated
- EditorProgramGUI → FileSystem: writeFile(path, buffer)
- FileSystem --> EditorProgramGUI: saveResult(success)
- EditorProgramGUI → RenderWindow: refreshStatus()

3.3. Database Model

3.3.1. Conceptual Model

The model contains the definition of entities and relationships among them in terms of the database (persistence). You should present it as a diagram and description of key elements in natural language.

The conceptual model describes the information that the application will store across sessions. Aside from the settings, this information revolves around keeping track of the user's session to allow for a seamless continuation when closing and restarting the application. The diagram is written using Chen notation in an ERD.
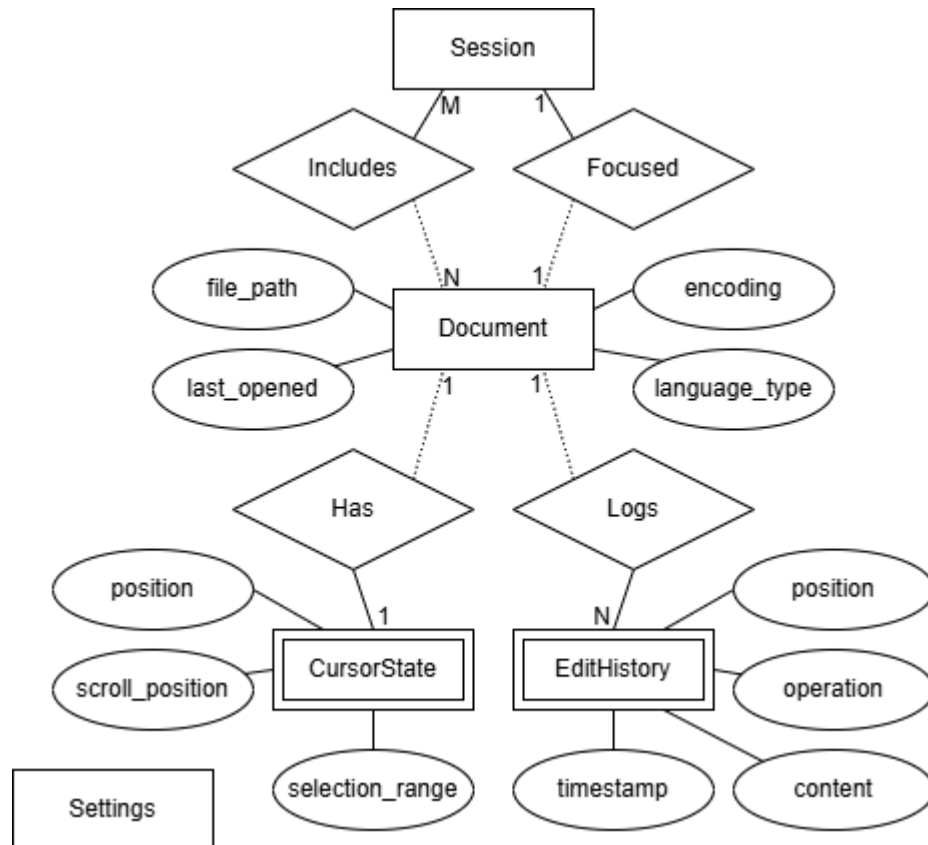

The application stores information about documents that have been opened such as their location, time of access, encoding, and language type. This allows the application to suggest recent files and open past files with custom remembered encodings or language types. Furthermore, the state of the cursor and some edit history can be saved optionally for each document to allow for resuming work at the same position and state. Lastly, a session can be saved when opening one or more documents at once to immediately open all those files upon opening the application.

3.3.2. Physical Model

The model presented as a diagram of the database that constitutes the implementation of the conceptual model in the selected database management system.

It is required to include an elaboration on the transformation of the conceptual model to the physical model. Description of transformation should include how elements like, e.g. inheritance and associations or other sophisticated modelling concepts have been handled.

The following diagram is an ERD written in Crow's foot notation. From the conceptual model, each entity has become a table with each attribute as a row and an added primary key id. Rows that are required for the functionality of the table are set as non-nullable. Each relationship has been represented as a relationship with a foreign key in the mandatory participation side aside from the singular many-to-many relationship. The SessionDocument is a new table used to represent the many-to-many relationship between Session and Document and uses each of their ids to create a composite key.
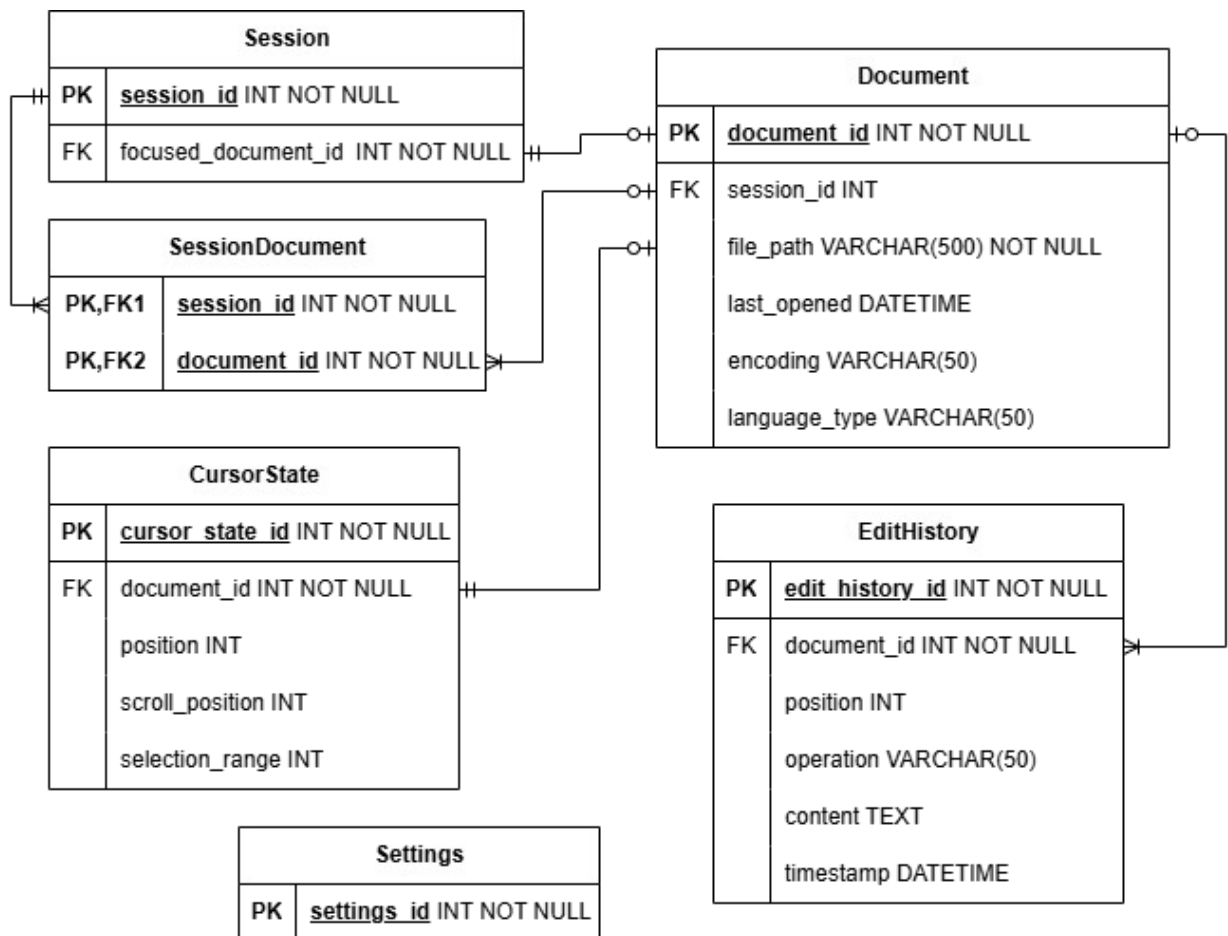
3.4. User Interface Design

In this section, you should present the distribution of <u>user interface components</u> with <u>events</u> and their assignments to the <u>behaviour specifications</u>, which these elements initiate or take part.

The application will consist of two pages, the main window where most of the functionality is, and the settings windows where various parameters and settings within the application can be adjusted. The general layout is shown in the diagram below.

Each UI component is represented as a rectangle while each event is represented as a rounded rectangle. Behavior specifications are written in each event box. Arrows are used to show association between ui components and events, show example layouts, and show navigation between pages. The example layouts show a single possibility for the application but there will likely be differences in the completed application such as more buttons or more settings.