Brief and Summary of the Software Project

Group: Yu Li 李渔, Xiangyu Kang 康翔宇, Yinghao Zhou 周瑛豪

Brief of project

The project aims to design and construct a new system for organizers in various fields, enabling them to create flexible templates for judging and grading. The system also includes modules for users, typically teachers, to assign marks to tasks and provide feedback to students regarding their homework or exams.

The entire project is divided into three interconnected and sequential parts: one for organizers, one for markers, and one for students who receive the results. To facilitate the workflow, two data formats, .toMark and .Mark, are predefined for communication between each pair of parts.

The first part, designed for organizers, is implemented by Yu Li using C++ with Qt. The second part, tailored for markers, is completed by Xiangyu Kang in Python. The final part, developed for students, is carried out by Yinghao Zhou, also in Python. Each part features an intuitive user interface along with robust back-end logic.

Details of implementing

Part for organizers

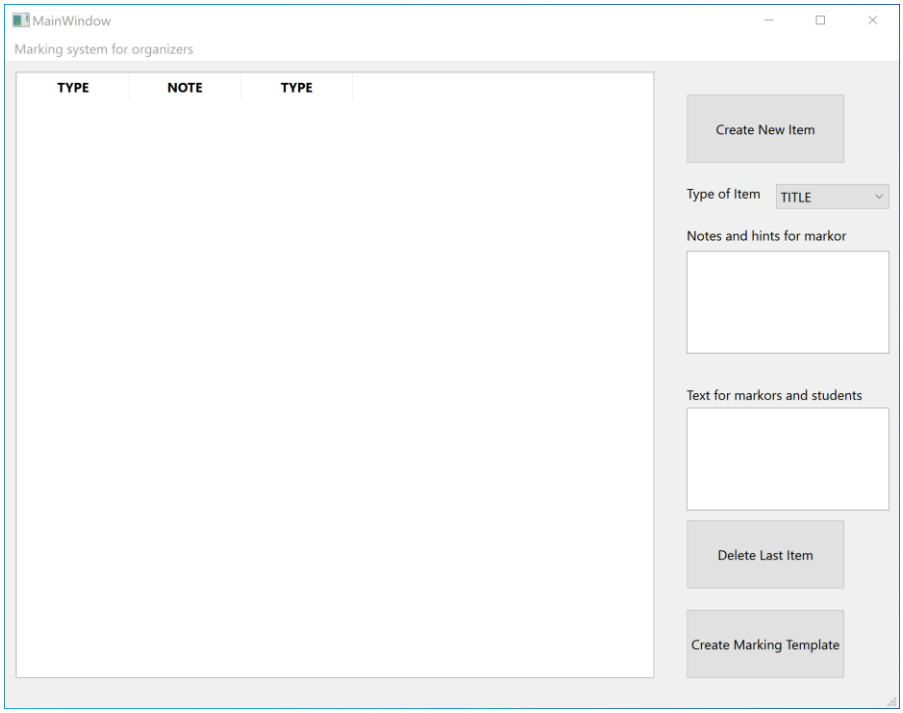
The first part of the project focuses on developing a customizable feedback system, with its primary component, EDIT.exe, serving as a tool for creating structured grading templates. This tool streamlines feedback by providing educators with a flexible and user-friendly interface.

In EDIT.exe, emphasis is placed on creating an intuitive interface that simplifies the process of building grading templates. Developed using the Qt framework with C++, the tool ensures reliability and cross-platform compatibility. The interface includes buttons for adding components such as titles, grading fields, and comment sections, dynamically generating items in the database. These items are displayed in a table along with their attributes, providing real-time feedback to the user. To maintain clarity, each component includes attributes like TYPE, NOTE, and TEXT, which define its purpose and content. A DELETE function allows editors to remove unwanted components. The finalized template is exported as a .toMark file, adhering to a standardized format that ensures compatibility.

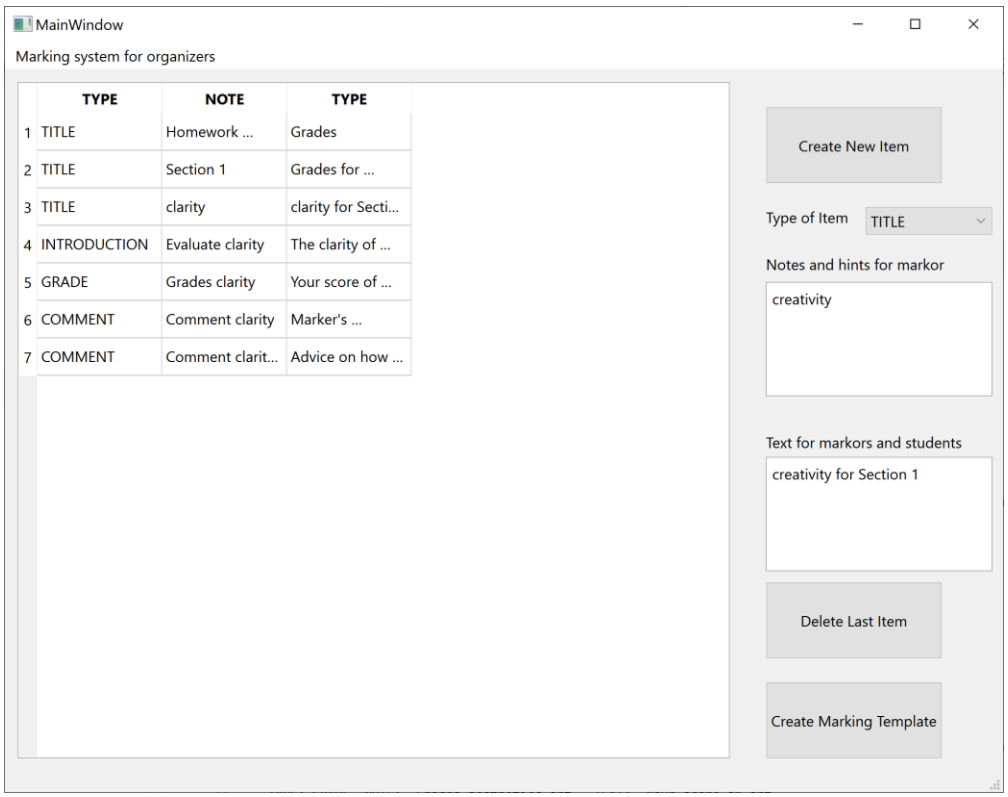
The .toMark file serves as a structured template for grading, bridging the workflow between EDIT.exe and subsequent processes. Each line defines a component with three attributes:

- TYPE: Specifies the type of component
- NOTE: Provides context or guidance for markers.
- TEXT: Displays content visible to users.

Example of EDIT.exe



Creating items



Result in output.toMark

```
TYPE: TITLE, NOTE: "Homework Grading", TEXT: "Grades"
TYPE: TITLE, NOTE: "Section 1", TEXT: "Grades for Section 1"
TYPE: TITLE, NOTE: "clarity", TEXT: "clarity for Section 1"
TYPE: INTRODUCTION, NOTE: "Evaluate clarity", TEXT: "The clarity of homework"
TYPE: GRADE, NOTE: "Grades clarity", TEXT: "Your score of clarity"
TYPE: COMMENT, NOTE: "Comment clarity", TEXT: "Marker's comment on clarity"
TYPE: COMMENT, NOTE: "Comment clarity improve", TEXT: "Advice on how to improve clarity"
TYPE: TITLE, NOTE: "creativity", TEXT: "creativity for Section 1"

```

Part for markers

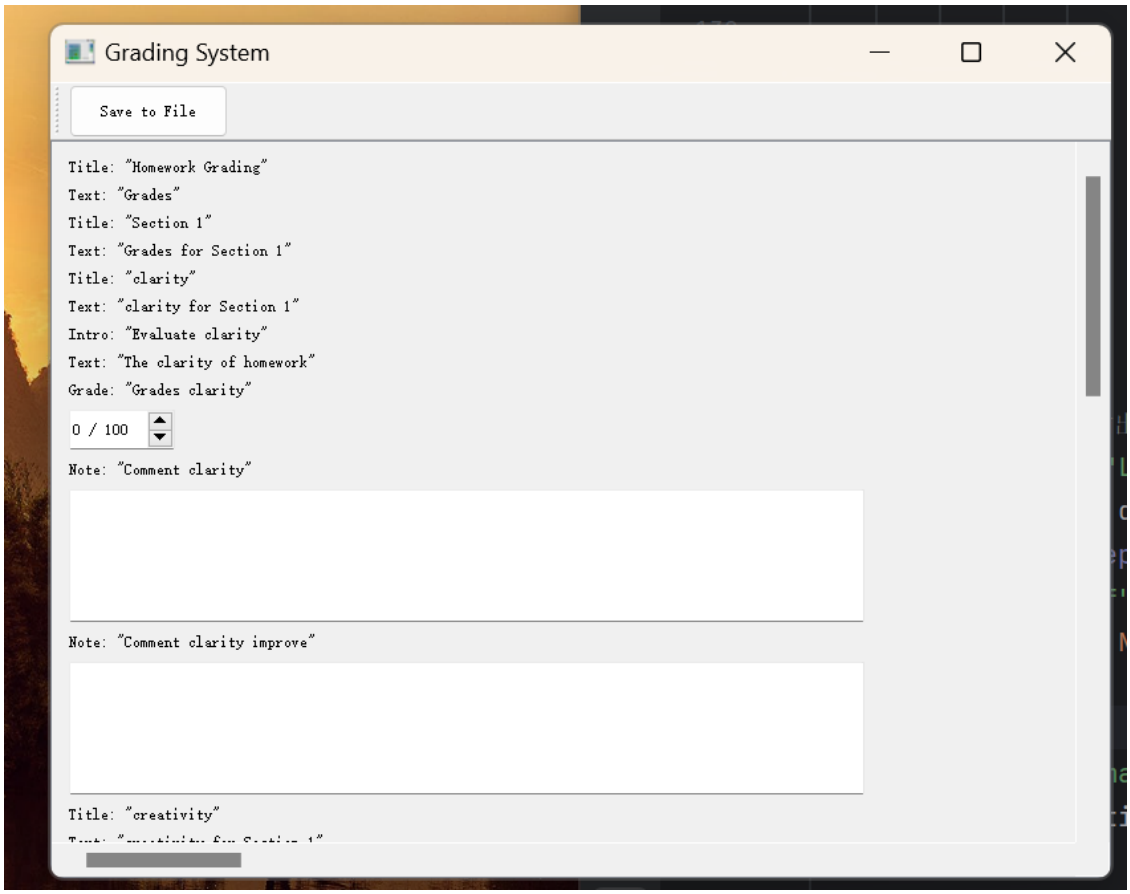
In the project's middle section, our team ensured that the data generated in the first section was correctly formatted and prepared for use in the third section. We managed various data inputs, including "title," "intro," "grade," and "comment," using methods like handle_title, handle_intro, handle_grade, and handle_comment within the GradingApp class. These methods leveraged PyQt5 widgets, such as QLabel, QSpinBox, and QTextEdit, to efficiently display and capture user input.

For example, when handling the "grade" input, a QSpinBox was used to allow users to select a score. Initially, the maximum value was set too high, leading to invalid entries. After discussions with the first-section team, the range was adjusted to ensure valid grades could be selected.

Similarly, the "comment" section’s narrow text box made it difficult for users to input detailed feedback. Resizing the QTextEdit resolved this issue, enhancing usability and readability.

Once the inputs were correctly formatted and adjusted, the team ensured that all data was accurately saved using the save_to_file method. For instance, grades were stored as TYPE: GRADE, NOTE: <note>, VALUE: <value>, while comments were saved as TYPE: COMMENT, NOTE: <note>, COMMENT: <comment>. This structured output facilitated seamless use in the third section, ensuring a smooth transition of information throughout the project.

Instance of Grading System



Part for students feedbacks

The student module is designed to display the content of files generated by the teacher grading module. After careful planning, we adopted the .txt file format for files produced by the teacher module due to its lightweight, widely supported, and easy-to-parse nature. To ensure smooth integration between the teacher and student modules, we defined the structure and content of the .txt files. These files include elements such as TYPE: GRADE, TYPE: COMMENT, and other formatted data. The student module reads, parses, and clearly displays this information to students.

To build the user interface for this module, we chose PyQt5 as the framework due to its flexibility and extensive features. For handling the display of long text content, we implemented QScrollArea, allowing students to scroll through content efficiently. Additionally, we used QVBoxLayout to create a layout system that automatically adjusts and displays each parsed line of content in an organized manner. This ensures even large files can be displayed successfully.

Instance of Student Feedback

```
TYPE: TITLE, NOTE: Homework Grading, TEXT: Grades
TYPE: TITLE, NOTE: Section 1, TEXT: Grades for Section 1
TYPE: TITLE, NOTE: clarity, TEXT: clarity for Section 1
TYPE: INTRODUCTION, NOTE: Evaluate clarity, TEXT: The clarity of homework
TYPE: GRADE, NOTE: Grades clarity, TEXT: Your score of clarity, SCORE: 12
TYPE: COMMENT, NOTE: Comment clarity, TEXT: Marker's comment on clarity, COMMENT: 1231231231
```

Conclusion

In summary, this project addresses key challenges in grading and feedback processes by leveraging modern technologies to create practical and efficient solutions. This tool empowers educators to create standardized and structured templates, while the student module ensures that feedback is accessible and easy to understand. By enhancing clarity and efficiency, this system not only improves the teaching and learning experience but also sets a foundation for scalable and adaptable grading solutions in diverse educational settings.