Design Document

Summary

Tomeo is a video playback and editing software designed for sportsmen, which is a lightweight video playback software. The functions are practical and easy to use, which can bring users an efficient use experience. And realized the overall design of human-computer interaction, operation logic, and beautiful interface. The five members of our team will start the development on November 22, 2021. The members are Wang Leshan, Li Xinyu, Xing Yufei, Zhang Yufan, and Chen Taiyu. First two of the members are responsible for the maintenance and writing of the software front-end, and the other three members are responsible for the development of the software back-end program.

During the entire software design process, we held five important meetings to determine the team leader and software development cycle issues, which will be described in detail in the appendix at the end of the article. The development cycle is generally divided into five cycles.

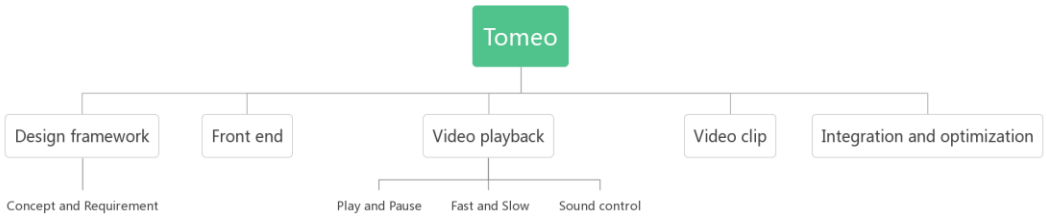


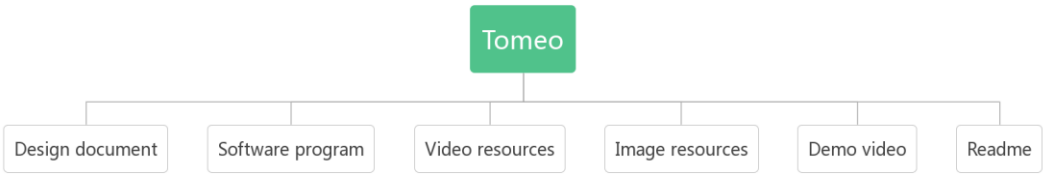
Figure 1

We spend a whole cycle (Cycle 1) to frame design and demand analysis of the project. Cycle 2.3.4 are the key development parts of the software. As can be seen from Figure 1, cycle 2 is to write the front-end layout of the software. The user interface maintains a consistent interface style, using consistent terminology, consistent steps, and consistent actions, allowing users to always think and operate in the same way. It also makes the software easy to use and efficient, attracting users to be more willing to use it and accept it. Our user interface is composed of several interface elements, but not all elements are equally important, ensuring that important elements appear to the user as soon as possible. Important or frequently visited elements are placed in a prominent position,

while less important or frequently used elements are placed in a less prominent position. And keep the user interface simple and clear and use colors reasonably to increase visual appeal. Choose a font that can be read smoothly on different resolutions and different types of screens. In addition, the consistency of the font is also maintained. When designing the interface, carefully analyze the functions of various elements and the logical relationships between the elements, and logically group them according to functions or relationships, and use controls to organize them to strengthen the links between the elements. Hiding the useless elements in the current state cannot only make the interface clean and clear, highlight the key points, but also ensure that users do not do invalid "work".

Cycle 3 implements the most important functions of the software, playing the selected video and completing the playback and pause operations of the video; mute, increase and decrease the volume, fast and slow play speed. Cycle 4 implements the editing feature that need to be performed on the selected video, which realizes Tomeo's function as a video editor, placing the videos that need to be edited in a selected queue. The clips inside the queue can be added or removed.

Figure 2 shows the composition of the entire project.



PACT Analysis and User Scenarios

First, the analysis of the interactive design framework PACT of the video player Tomeo designed by our group. For the basic interaction design framework, designers use scenarios and requirements to create user interfaces. Tomeo is a video player that can meet the needs of PC and PE at the same time. Users interact through various buttons on the interface. Under more specific PACT analysis, people are the ultimate perceiver of interaction no matter what kind of design we make, it is ultimately used by people, and people have always been at the center of the design. We have considered the use of

most special groups, ensured the high recognition of the design, and highlighted all functional buttons. The extremely high interactivity and ease of use can make people familiar with his software usage rules in a short period of time. Regardless of the user's recognition level, they will easily learn to use it.

Tomeo was originally designed as a portable video player and editor designed for exercisers. When a climber shoots a video with sports photography equipment such as GoPro, Tomeo can not only be used as a very reliable video player to complete its most basic video playback function, but also can edit and splice the video with other videos to achieve a simple and convenient video editor function. We believe this will be deeply loved by sports enthusiasts such as mountaineering. This is the first use-case scenario among them.

Another use-case scenario is that when users need to edit and merge multiple videos, they can add them to the operation list in the upper right corner of the software by clicking on the video image that needs to be operated, and then after merging, End the merging operation of multiple videos by clicking the Finish button. This is Main Success Scenario, but there are some Alternate Scenarios. If the user needs to operate too many videos, the system will prompt the user to delete the number of videos and do so in multiple times. If you do not add a video to the operation list, you will not be able to click the operation complete button, and the player will play the original video.

The third use-case scenario is that the user uses Tomeo to play the video. The Main Success Scenario is that the user selects the video that needs to be played, and performs normal half-speed, double-speed playback of the video, and adds and subtracts the volume of the video. Mute operation. Alternate Scenarios means that if the user does not select the corresponding video, the player will not play the video. If the software does not support the user's video type, the system will prompt.

Platform

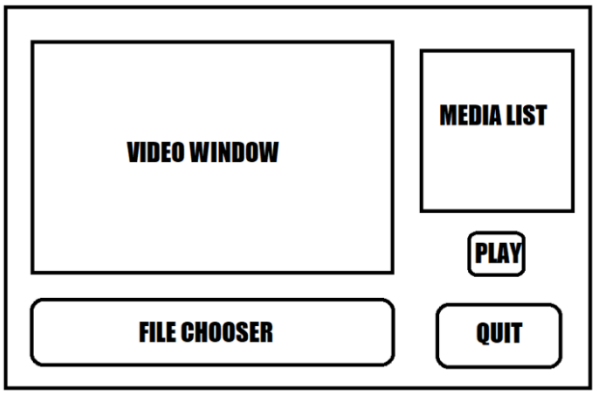
For Tomeo's design platform, all our development take place on the Desktop with Qt and C++, which includes both the front and back ends. We mainly design for all PC users. The responsive design allows the software to perform well on most ports. This will greatly increase the audience of the software.

Cycle 1:

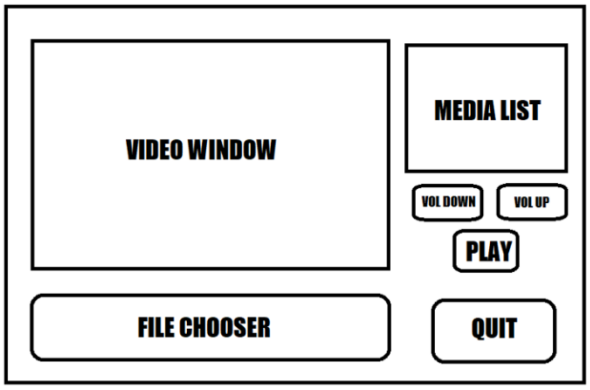
Basic Concept and Front-end Design

Following the requirements of coursework 3, the team had conducted several meetings to discuss, analysis and set basic development goals for the project. According to requirements and real-life products such as VLC Player, the prototype was decided to have both features from video player and video editor. The project will be able to perform basic operations on videos, for example: play/pause, volume adjustment, speed control, etc. Also be able to add(remove) clips to create a new video. This cycle is extremely important because it is the foundation of the whole development process. Without a well-considered plan, the final product can be disastrous or there won't be any outcome at all.

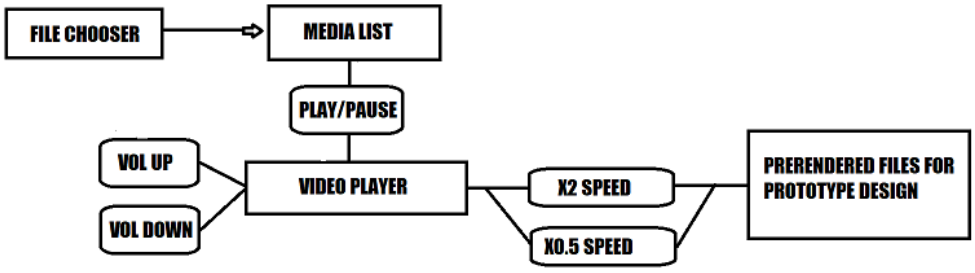
The prototyping is mostly done by sketching using Microsoft Paint software, which is a simple yet useful tool, it can be easily shared and modified. During the first cycle, the team came up with several design ideas, with consideration of time and skill, the one shown below is decided to be the main layout.



After the initial design, volume controls were added immediately to suit the feature logical design.



The basic features were designed logically in the first cycle. Shown by the model, the video clips are selected by using the file chooser and added into a media list, which will also be shown on the front as acknowledgement. The play/pause button will operate video, and two separate buttons will control volume level. Speed controls were added later to complete the features. As for a prototype, the team decided to use prerendered videos to simulate real speed up/slow down actions.



Codes:

The first cycle didn't contain any coding level actions. All prototyping were done at logical level. The actual coding starts from the second cycle.

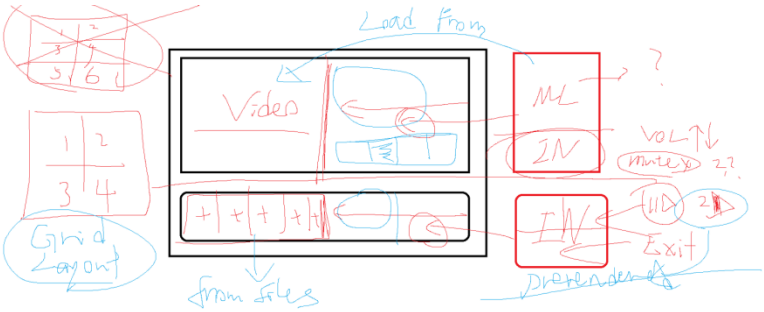
Evaluation:

In the first cycle, heuristic evaluation was used to help evaluate the design.

Heuristic evaluation was conducted first in order to gather as much ideas as possible. After analyzing the requirements, key features were set, each teammate was then asked to observe, research, and design their own layout for frontend. By doing this, the team was able to think independently and came up with different but reasonable designs. Since accessibility is the main idea of frontend design, heuristic evaluation will be fair to

discover a commonly accessible layout. After individual evaluation, team members shared their own designs, and they were finally mixed to become the designated layout shown before.

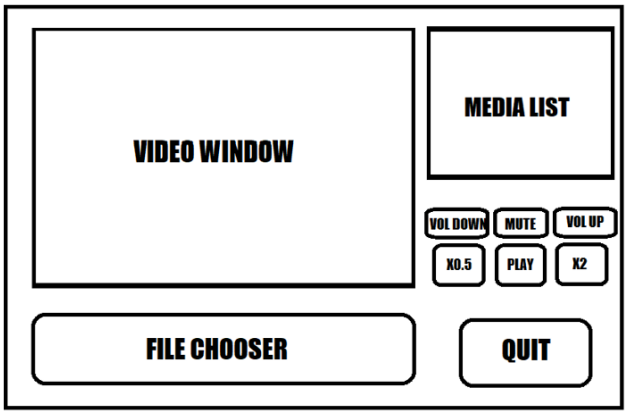
Below is the integration of designs the team did in this cycle, many of the designs were changed and improved in later cycles.



Cycle 2:

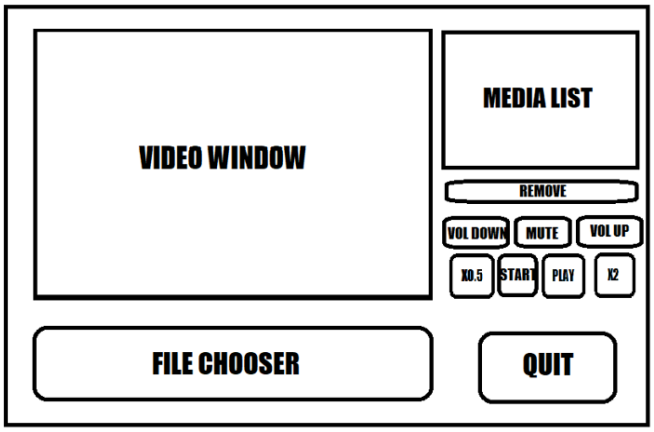
Front-end Realization and Improvements

The second, third and fourth cycle were going parallel with each other for best efficiency. The second cycle focus on further designs and realization of the project's front-end. The goal is to create a complete, functional and accessible interaction window. After the layout was set in the first cycle, the codes provided by teacher were carefully checked and great number of changes were made. Since both front-end and back-end programming were ongoing at the same time, frequent meetings and progress sharing were conducted to get information and make changes. Sketching and C++ programing are the main techniques used in this cycle, with support of Microsoft Paint and QT. The designs changed greatly due to back-end features adjustments and improvements, which are shown below:

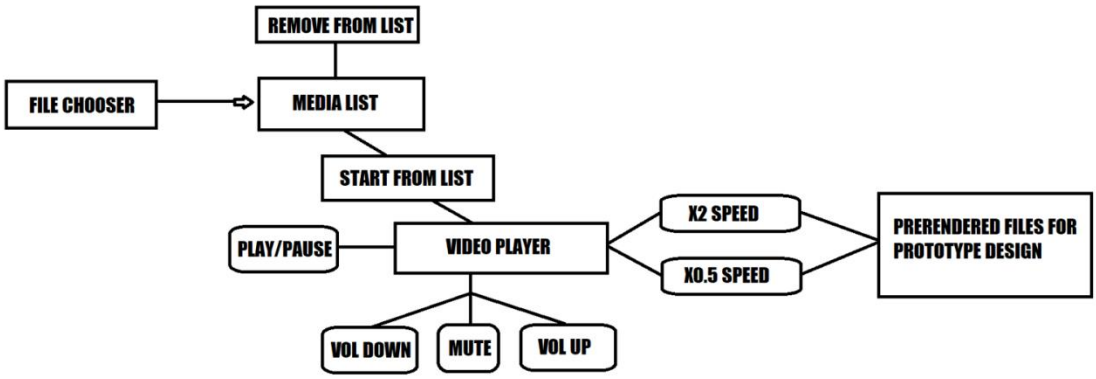


Speed up and slow down features were implemented, thus two more buttons were added accordingly, placed beside the play/pause button as a typical layout design. Also

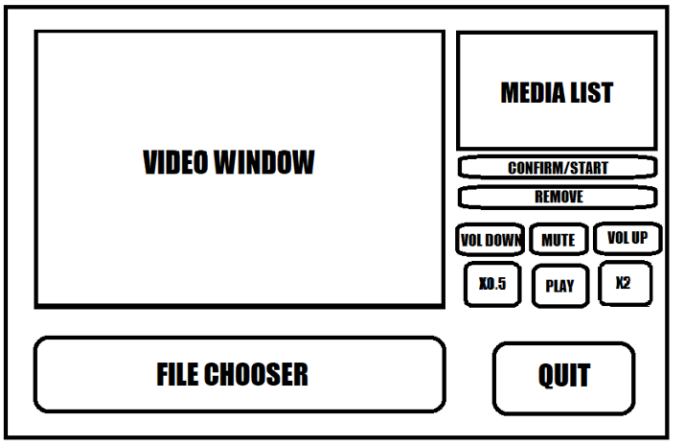
the mute function was added in the third cycle, therefore a mute button was added between volume increase and volume decrease buttons.



More changes were made as back-end development faced problems. The original design used play/button to control video, however when implementing the feature, team members conducting cycle 3 and cycle 4 faced technical problems and was unsolvable in a short time. To keep up with plan, the team decided to adept an alternative way, that is to add an additional button to start playing videos from the media list, after that use the original play/pause button to control the video. This method turned out to be achievable. Also, the remove button was added to remove video clips from media list as a completion of editing features. The updated feature model shows the change of logic process of the project.



The final layout considered accessibility and consistency, layout was improved as the button to play video from media list was placed above the remove button.



Codes:

In coding level, the major changes were layout designs and additional buttons. The original design was using QHBoxLayout, the improvements were made to change it to QGridLayout. Addition buttons were added according to the design, proper spacing, icons, fonts and colors were implemented for beauty and accessibility.

(Partial codes as example...)

```
//Customized buttons
QPushButton quit("Quit");
quit.setIconSize(QSize(60,60));
quit.setIcon(QIcon("../image/qu.jpg"));
quit.setMinimumSize(100,100);
quit.setStyleSheet(list.join(';'));
quit.setFont(QFont("Showcard Gothic", 20));
QObject::connect(&quit,SIGNAL(clicked()),&app,SLOT(quit()));

QWidget *buttonWidget2 = new QWidget();
QHBoxLayout *layout2 = new QHBoxLayout();
buttonWidget2->setLayout(layout2);

QPushButton slow("");
slow.setIconSize(QSize(50,50));
slow.setIcon(QIcon("../image/s.jpg"));
slow.setFont(QFont("Showcard Gothic", 12));
slow.setStyleSheet("background-color: white;");
layout2->addWidget(&slow,0,Qt::AlignBottom);

QPushButton play("Play/Pause");
play.setIconSize(QSize(50,50));
play.setIcon(QIcon("../image/p.jpg"));
play.setFont(QFont("Showcard Gothic", 12));
play.setStyleSheet("background-color:white;");
layout2->addWidget(&play,0,Qt::AlignBottom);

QPushButton fast("");
fast.setIconSize(QSize(50,50));
fast.setIcon(QIcon("../image/f.jpg"));
fast.setFont(QFont("Showcard Gothic", 12));
fast.setStyleSheet("background-color: white;");
layout2->addWidget(&fast,0,Qt::AlignBottom);

QWidget *buttonWidget3 = new QWidget();
QHBoxLayout *layout3 = new QHBoxLayout();
buttonWidget3->setLayout(layout3);
QPushButton vd("");
QObject::connect(&vd,SIGNAL(clicked()),player,SLOT(volumnDown()));
vd.setIconSize(QSize(120,30));
vd.setIcon(QIcon("../image/vd.jpg"));
vd.setFont(QFont("Showcard Gothic", 10));
vd.setStyleSheet("background-color: white;");
layout3->addWidget(&vd,0,Qt::AlignBottom);
```

The changes between prototype and actual implementation were mentioned before, a list is provided here for a clearer view.

Feature/Design	Prototype	Implementation	cause
Play/Pause	Control video	An additional button to start playing	Technical difficualties, time consuming
Layout	/	Remove button and start button were added	Changes were made in back-end design

Evaluation:

In the second cycle, cognitive walkthrough was used to help evaluate the design.

Cognitive walkthrough was conducted because front-end design is directly accessed by users, it must be highly reasonable and accessible. All features should be displayed and recognized by users.

Firstly, target users are identified in early cycle, which are outdoor enthusiasts who wish to explore and organize a large personal video library. They would use this program for video clips editing. Therefore, a task is generated, which is to create a video with several clips, and the final product will be able to play in different volume and speed. After task creation, action sequence is designed. The typical sequence would be to add clips by using file chooser, after adding, the file names will appear in the media list. Users can remove clips by clicking remove button. After creation, a start/confirm button is clicked to generate the final product and start playing. During playing, users can adjust volume, playback speed and play/pause at real-time. Then, several walkthrough of action sequence were conducted by team members and testers to gather feedbacks. Finally, according to feedbacks, further improvements were made on button size, font style, background color and spacing for better accessability and interaction experience.

The crucial feedbacks were recorded in a form.

Feedback	Improvement
Play button too small	Resize
Hesitation after editing	Place the start button above remove button
Font is ugly?	Showcard Gothic font
Low contrast	Set background color to be energrtic green
Pressing wrong button	Enlarge spacing
Video window size?	No change

Cycle 3:

Basic Features Realization

The third cycle focuses on the design and implementation of the back-end functions of the project. The goal of this cycle is to realize various functions of video playback, pause/play, 2x and 0.5x speed play and volume adjustment. As the most important function of video player, these functions are most closely related to users. The rationality and practicability of playback function is the key of the project.

After combining the layout designed in the second cycle, the main work here is to assign corresponding functions to the buttons through connect(). All these works are completed rely on QT.

First, pause the play, due to design errors, the video can only be suspended at the beginning and cannot be replayed again. The second-generation solution can solve this problem perfectly.

```
void ThePlayer::start(){
    if(!mTimer->isActive()){
        mTimer->start();
        play();
    }
}

void ThePlayer::pause(){

        if (mTimer->isActive()){
            mTimer->stop();
            pause();

    }
}
```

(Original version)

```
void ThePlayer::start(){
    if (flag == 0){
        if (mTimer->isActive()){
            mTimer->stop();
            pause();
            flag = 1;
        }
    }
    else{
        if(!mTimer->isActive()){
            mTimer->start();
            play();
            flag = 0;
        }
    }
}
```

(Final version)

The 2x button and 0.5 button in the main interface are given the function of playing speed doubling video.

```
void ThePlayer::f(){
    QUrl u;
    u.setUrl("../videos/g.wmv");
    setMedia(u);
    play();
}

void ThePlayer::s(){
    QUrl u;
    u.setUrl("../videos/g1.wmv");
    setMedia(u);
    play();
}
```

The last is the volume adjustment function. Firstly, it realizes the functions of increasing muscle volume and reducing volume, and then obtains relevant knowledge from the Internet to realize the mute function.

```
void ThePlayer::volumnUp(){
    if(i<=90){
        i = i+10;
        setVolume(i);
    }
}

void ThePlayer::volumnDown(){
    if(i>=10){
    i = i-10;
    setVolume(i);
    }
}
```

(Original version)

```
void ThePlayer::volumnUp(){
    if(i<=90){
        i = i+10;
        setVolume(i);
    }
}

void ThePlayer::mute(){
        setVolume(0);
}

void ThePlayer::volumnDown(){
    if(i>=10){
    i = i-10;
    setVolume(i);
    }
}
```

(Final version)

Codes:

At the encoding level, the main change is to combine pause and play into one button. The original design used two buttons to realize the pause and playback functions, and the improved design changed it to one button.

Function	Volume	Play/Pause	2x/0.5x play
Difference	In the initial design, the volume adjustment is more flexible and accurate. The volume value of the implementation version can only reach several values	This function, the preliminary design of this function has been implemented and can work well.	The preliminary design of this function is that the player can switch the playback speed of video at any time. The final function is to play different speed versions of the saved video.
Cause	Time limit		Technical limit

Evaluation:

The evaluation technique used there is cognitive walkthrough.

This method can make specific evaluation for the functions implemented in cycle. And this evaluation method can well verify the rationality of the implementation of these functions, which is also the most important part of these functions. The project must ensure that these functions closely related to users' use are humanized and users can use them easily.

The results of the assessment were positive. By analyzing users' psychology, users can use these functions well and get the feedback they want. The realized functions are determined to be reasonable after evaluation. Although there are some differences from the original design, and some aspects are still worthy of optimization, the overall evaluation results show that this part of the function is reasonable and can be used. The changes made to the project in this cycle are also accepted. These changes also make the function of the program more in line with the actual needs, more humanized and more feasible.

Function	Volume	Play/Pause	2x/0.5x play
Evaluation result	The operation is convenient and simple, and the volume changes normally. The volume value should be more accurate.	The operation is convenient and simple. Users can pause and replay the video.	Users can easily play videos at different speeds, but users want to change the playback speed at any time. This is the feedback users want more.

Cycle 4:

Editing Feature Realization

The fourth cycle is going to achieve the info, initial start of the video play, add more than one videos into a queue and continue to play them, delete a video in the queue for not playing it. This cycle is very important, and it was selected as the highest priority because it realizes the most basic video playback function of the program and selects the video to join the queue for continuous playback. In the fourth cycle, the playlist in QMediaPlaylist class is used. It is used to add the URLs of multiple videos to a media to be played. Then QT will parse the URLs of videos and play them in sequence accordingly.

In the prototype, when a video is selected, the connect function will call the corresponding add function to add it to a playlist. When adding, the info in the upper right corner will also be realized, and the file name of the selected video will be added accordingly. When user finishes adding, he can play all the videos in the list by clicking the start button. He can also delete a video in the last of the list if he wants to.

The playlist is chosen because it can be easily implemented, continuous playback can be realized through the URL of the video.

Codes

In the coding level of the fourth cycle, it has many differences between the prototype and the implementation. The first difference is about the function ‘info’. In the prototype the info function is realized by changing the thumbnail, When a video is selected, its thumbnail is changed to a picture with a tick in the lower right corner. However, in the real implementation, because of technical difficulty, it is changed to the display its file name on the listwidget by choosing video, the listwidget is belong to the QListWidget class. The second difference is about the function ‘delete’. In the prototype the delete function is to clear the entire video list, all the videos will be deleted. However, user may not want to delete all videos, they may just want to delete some videos. So in the real implementation, the delete function is to delete the last video in the video list.

(Partial codes as example...)

```
void ThePlayer::playul () {
    QMediaPlaylist *playlist = new QMediaPlaylist();
    for (int i = 0;i < countt;i++)
    {
        if(ulist.at(i).isEmpty() == false)
            {
                playlist->addMedia(QMediaContent(ulist.at(i)));
                cout << i << endl;
            }
    }
    setPlaylist(playlist);
    play();
}
```

Evaluation

In the fourth cycle, it uses Heuristic Evaluation, it is a kind of evaluation that let everyone evaluate according to the criteria, list the possible problems, and finally focus on the discussion.

This evaluation technique was used because the fourth cycle is a very important cycle. It has many basic functions such as playing videos. Therefore, in order to make everyone in the group understand and be satisfied with it, heuristic evaluation is necessary. Everyone in the group checks whether the final implementation meets their

expectations and raises questions or areas for improvement in the discussion to achieve a better design.

The outcome of the evaluation is that the design meets the expectation, but still needs to be improved. This is because of the technical difficulty; some better functions cannot be implemented in a limited time. And the change of the cycle is accepted because of the limited time.

Below is the evidence of the evaluation

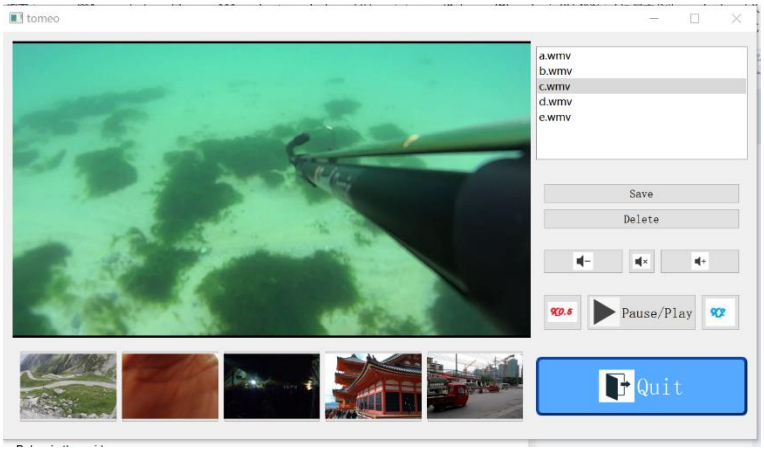
Feedback	Improvement
A video can not be added twice	Change the way of adding videos
Thumbnails are not easy to achieve	The information will be printed on the window
Delete all videos in the list is not humanitarian	Delete will only remove the last video in the list

Cycle 5:

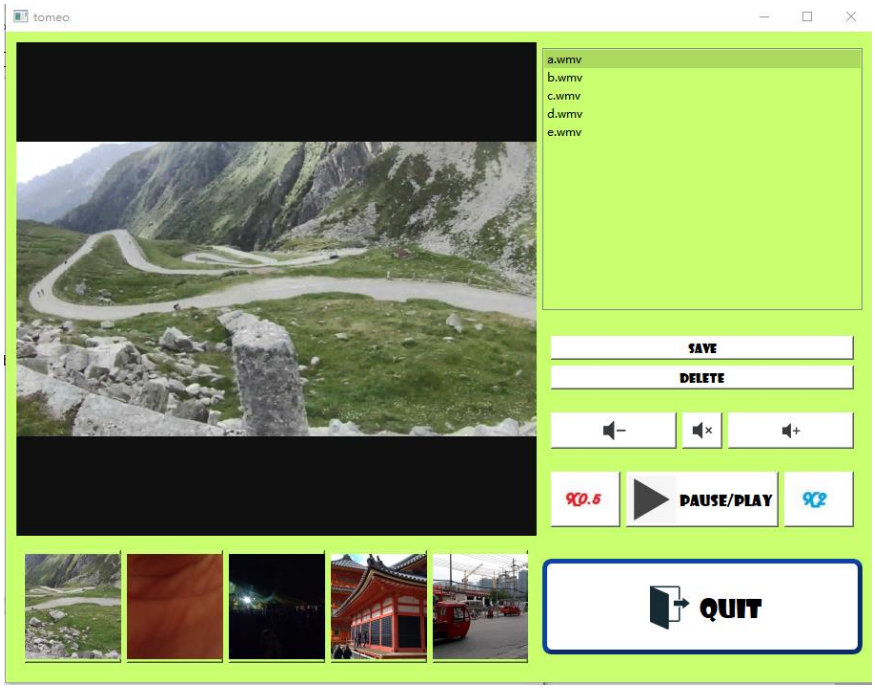
Finalization

The fifth cycle focuses on summarizing the combination of front-end and back-end to produce a complete project. Our goal is to achieve all the functions we expect. After the fourth cycle, we have all the front-end and back-end code we need. So far, we have basically mastered and implemented a GUI application. In this cycle, meetings are often held to ensure the implementation of functions and debug some bugs. C + + programming is the main technology used in this cycle and supports QT.

The original design is relatively lack of color.



The final design is to further optimize the interface and use motion green to enhance the visual effect of the interface.



Codes:

At the code level, the connect () function in QObject is mainly used to establish the connection between the signal and the slot, so as to realize the connection between the front-end button and the corresponding function.

(Partial codes as example...)

```
..
QPushButton quit("Quit");
quit.setIconSize(QSize(60,60));
quit.setIcon(QIcon("../image/qu.jpg"));
quit.setMinimumSize(100,100);
quit.setStyleSheet(list.join(';'));
quit.setFont(QFont("Showcard Gothic", 20));
QObject::connect(&quit,SIGNAL(clicked()),&app,SLOT(quit()));

QWidget *buttonWidget2 = new QWidget();
QHBoxLayout *layout2 = new QHBoxLayout();
buttonWidget2->setLayout(layout2);

QPushButton slow("");
QObject::connect(&slow,SIGNAL(clicked()),player,SLOT(s()));
slow.setIconSize(QSize(50,50));
slow.setIcon(QIcon("../image/s.jpg"));
slow.setFont(QFont("Showcard Gothic", 12));
slow.setStyleSheet("background-color: white;");
layout2->addWidget(&slow,0,Qt::AlignBottom);

QPushButton Pause("Pause/Play");
QObject::connect(&Pause,SIGNAL(clicked()),player,SLOT(start()));
Pause.setIconSize(QSize(50,50));
Pause.setIcon(QIcon("../image/p.jpg"));
Pause.setFont(QFont("Showcard Gothic", 12));
Pause.setStyleSheet("background-color: white;");
layout2->addWidget(&Pause,0,Qt::AlignBottom);

QPushButton fast("");
QObject::connect(&fast,SIGNAL(clicked()),player,SLOT(f()));
fast.setIconSize(QSize(50,50));
fast.setIcon(QIcon("../image/f.jpg"));
fast.setFont(QFont("Showcard Gothic", 12));
fast.setStyleSheet("background-color: white;");
layout2->addWidget(&fast,0,Qt::AlignBottom);
```

```
QWidget *buttonWidget3 = new QWidget();
QHBoxLayout *layout3 = new QHBoxLayout();
buttonWidget3->setLayout(layout3);
QPushButton vd("");
QObject::connect(&vd,SIGNAL(clicked()),player,SLOT(volumnDown()));
vd.setIconSize(QSize(120,30));
vd.setIcon(QIcon("../image/vd.jpg"));
vd.setFont(QFont("Showcard Gothic", 10));
vd.setStyleSheet("background-color: white;");
layout3->addWidget(&vd,0,Qt::AlignBottom);

QPushButton mu("");
QObject::connect(&mu,SIGNAL(clicked()),player,SLOT(mute()));
mu.setIconSize(QSize(30,30));
mu.setIcon(QIcon("../image/vm.jpg"));
mu.setFont(QFont("Showcard Gothic", 10));
mu.setStyleSheet("background-color: white;");
layout3->addWidget(&mu,0,Qt::AlignBottom);

QPushButton vu("");
QObject::connect(&vu,SIGNAL(clicked()),player,SLOT(volumnUp()));
vu.setIconSize(QSize(120,30));
vu.setIcon(QIcon("../image/vu.jpg"));
vu.setFont(QFont("Showcard Gothic", 10));
vu.setStyleSheet("background-color: white;");
layout3->addWidget(&vu,0,Qt::AlignBottom);

QWidget *buttonWidget4 = new QWidget();
QVBoxLayout *layout4 = new QVBoxLayout();
buttonWidget4->setLayout(layout4);

QPushButton sv("Save");
QObject::connect(&sv,SIGNAL(clicked()),player,SLOT(playul()));
sv.setFont(QFont("Showcard Gothic", 10));
sv.setStyleSheet("background-color: white;");
layout4->addWidget(&sv,0,Qt::AlignBottom);

QPushButton dl("Delete");
QObject::connect(&dl,SIGNAL(clicked()),player,SLOT(dl()));
dl.setFont(QFont("Showcard Gothic", 10));
```

Evaluation:

In the fifth cycle, user test is used to help evaluate the design. User test is conducted because the final output is for users. It must be usable, bug free and easy to use.

First, the preparation before the test was carried out. 30 users were recruited, and the data filled in by the recording tool was prepared. The second step is the testing stage. After introducing the functions of the software to the user, the user uses the software himself, and makes a simple evaluation on the availability of the software. It is divided into four levels: availability, slight availability, unavailability and availability disaster. The last step is data collation, which collates the data of all users into a table and summarizes the users.

The table of key information collected from the data is shown in the following table.

Evaluation grade	Availability	Slight availability	Unavailability	Availability disaster
number of votes	28	2	0	0

Ethical Statement

In this project, research on humans is taken under strict ethical guidance. Following the rules of the university regulations, all research participants include those taken surveys and given feedbacks were told about how any personal data collected about them will be used, stored, processed, transferred, and destroyed. No sensitive data was collected during the process. The research participants were asked and agreed to share their opinions and feedbacks with the team and anyone who might read this document.

Conclusion

The whole team made a detailed plan at the beginning of this development project. The plan strictly defines the time of each meeting and the working time of each cycle. The team can complete relevant work within the specified time, and all team members can attend the team meeting on time. At the meeting, all members reported their actions. The team members discuss the difficulties encountered and make necessary modifications and optimization to the later plan.

Throughout the development process. At the beginning, there was a problem of untimely information sharing in the group. There is duplication of work and out of sync progress among members. Later, after the group meeting, the members of the group increased the frequency and real-time of information sharing, and the work efficiency of the group

was greatly improved. In addition, many technical problems will be encountered in the development process, and the members of the group will discuss together to help the members with difficulties. Group discussion is not only the main way to solve problems, but also a very effective method.

In this development work, members of the group can complete their tasks well and are very responsible for the team and the project. Team members get along well, and each member has good communication skills. Everyone helped each other and the cooperation within the group was very successful.

For the results of this project, the initial design has been basically realized. The software has the basic functions of video player and reasonable layout and appearance. Users can use it easily. However, there are still many places in the software that can be optimized and improved. Limited by time and technology, more functions and optimization are not reflected in the results.

The group needs to learn from this experience, improve work efficiency and improve its own ability. And do better in the future work.

Meeting Record

Meeting	Content
1(11.22)	We Created a group and exchanged contact information. On this basis, after a long period of discussion and analysis, we established the research direction and established a shared code library GitHub and a shared process document WPS. And established a mode of multiple progress reports and exchanges every week. In this meeting, we formulated the requirements that need to be completed, and assigned each requirement to the individual.
2(11.28)	Discussed Tomeo's specific functions and implementation of components (including playback and pause, volume adjustment, playback speed, etc.). Arranged and planned the design and development time, and determined that the project roughly consists of two parts: code and documentation. Preliminarily divide the project into multiple cycles (layout-1, volume-1, video join list, play, slow play-3, integrated function-1, overall debug-1).
3(12.4)	The team members share the progress of some of their projects and communicate with them on the problems they have encountered. And decided to add a new function to save the contents of the list locally ---- save and delete. Everyone analyzed the progress of their cycle.
4(12.17)	Check the leaks of the entire project, analyze the problems and gains encountered in the process of project development and design, analyze the contribution of each person's cycle, and complete the final report.