CMPT 363 D100: USER INTERFFACE DESIGN

Group Project Part 3

Project Team 6

Student Name: Qi Dai, SFU Email: qi_dai@sfu.ca, SFU ID: 301363579
Student Name: Zeyong Jin, SFU Email: zeyongj@sfu.ca, SFU ID: 301353174
Student Name: Zhuo Liu, SFU Email: zla182@sfu.ca, SFU ID: 301385532
Student Name: Guiyi Su, SFU Email: guiyis@sfu.ca, SFU ID: 301300440
Student Name: Feifan Wang, SFU Email: feifanw@sfu.ca, SFU ID: 301328222

August 6th, 2021

Component 1: Cognitive Walkthrough

1.1: Context and scenario

During a regular semester of a university program, university students have to use a calendar system to import all courses at the beginning of the term, view schedule, add, modify or delete events during the term, and remove all courses at the end of term.

In a regular academic semester, Joe, a 21-year-old and year-four CS major student, takes three tough courses. Joe's schoolwork is very busy, with many deadlines and lectures every week. In his leisure time, Joe also needs to do some personal events like shopping. Joe wants to import events of all enrolled courses to the calendar at the beginning of the term and can view events on different dates during the term. He also needs to add, modify, delete different events. At the end of the term, he is likely to remove all courses' events from the calendar.

1.2: Potential/target user

University or college students, especially those freshmen who are not familiar with calendars and the students who have a heavy workload in each term.

1.3: Representative task for V-MFP & 1.4 Cognitive Walkthrough

In this representative task, Joe wants to import all events of enrolled courses at the beginning of the term. During the term, he needs to view the schedule of a certain date, say July 15th. He also wants to change displayed event types, when viewing schedules. When adding a new event, he wants to clear input, reverse operation, then accept the grammar suggestion from the system. When necessary, he needs to modify information of existing events, delete events. At the end of the term, he wants to remove all courses' events to make the calendar clear.

The team invites *Leah Fei* from *Team 13* to evaluate the vertical MFP, and the results are presented via a table as follows.

Action Sequence	**Does the user know what to do given the action?**	**Can the user find the right interface component to perform this action?**	**Can the user associate the feedback from the interface to the correct action they perform?**	**Does the user understand the feedback so that they know where they are in the task after performing the correct action?**	**Comment**
Action 1: **Click the right button to import course events.**	No, the user is a bit confused about this action, because a typical calendar does not have this feature.	Yes, the user finds out the correct component as it is distinguishable.	Yes, the user finds out that there are some course events on certain dates.	No, the user stays at the home page and is a bit confused about the current status.	The user is not very clear about the function of the button.

Action Sequence	Does the user know what to do given the action?	Can the user find the right interface component to perform this action?	Can the user associate the feedback from the interface to the correct action they perform?	Does the user understand the feedback so that they know where they are in the task after performing the correct action?	Comment
Action 2: **Click the right button to view the schedule for a certain day (e.g., July 15th).**	Yes, the user knows the action, because it is common.	No, the user cannot find out the component to do this action.	Yes, the user is satisfied with the feedback of the pop-up window.	Yes, the user stays at the pop-up window and knows where she is in the task.	The click button is not clear.
Action 3: **Click the right button to change Event Type.**	Yes, the user knows the action, because it is common.	No, the user does not realize there is a button.	Yes, the user is satisfied with the feedback of the pop-up window.	Yes, the user stays at the pop-up window and knows where she is in the task.	The click button is not clear.
Action 4: **Tell the system to add an event with an illegal time.**	Yes, the user knows the action, as it is common.	No, the user is a bit about the usage of the button.	Yes, the user is satisfied with the feedback of the warning message.	Yes, the user backs to the edit area of the home page and knows where she is in the task.	The user does not know whether the click button can be clicked.
Action 5: **Click the right button to clear all the events.**	Yes, the user knows the action, as it is common.	Yes, the user finds out the correct component, because it is clear.	Yes, the user finds the input area is set to default.	Yes, the user stays at the home page and knows where she is in the task.	No comment.
Action 6: **Click the right button to undo the recent operation.**	Yes, the user knows the action, as it is common.	Yes, the user finds out the correct component, because it is clear.	Yes, the user finds the last operation was reversed.	Yes, the user stays at the home page and knows where she is in the task.	No comment.

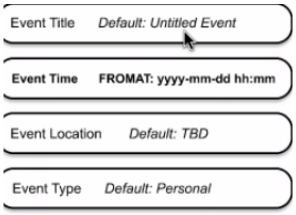
Action Sequence	Does the user know what to do given the action?	Can the user find the right interface component to perform this action?	Can the user associate the feedback from the interface to the correct action they perform?	Does the user understand the feedback so that they know where they are in the task after performing the correct action?	Comment
Action 7: **Tell the system to create a new event.**	Yes, the user knows the action, as it is common.	Yes, the user finds out the correct component, because it is obvious.	Yes, the user finds the new event on the input date.	Yes, the user backs to the home page and knows where she is in the task.	No comment.
Action 8: **Tell the system to automatically correct spelling errors.**	Yes, the user knows the action, as it is common.	Yes, the user finds out the correct component, because it is straightforward.	Yes, the user finds the advice from the system and accepts it.	Yes, the user backs to the home page and knows where she is in the task.	No comment.
Action 9: **Tell the system to display the event info.**	Yes, the user knows the action, as it is common.	No, the user does realize that the event is clickable.	Yes, the user is satisfied with the pop-up window.	Yes, the user stays at the pop-up window and knows where she is in the task.	The button seems not clickable.
Action 10: **Click the right button to edit an event.**	Yes, the user knows the action, as it is common.	No, the user is a bit confused about the expression of the button.	Yes, the user is satisfied with the pop-up window.	Yes, the user stays at the pop-up window and knows where she is in the task.	The click button is not clear.
Action 11: **Click the right button to delete an event**	Yes, the user knows the action, as it is common.	No, the user mistakes the button for decoration.	Yes, the user finds the event is deleted after confirmation.	Yes, the user stays at the pop-up window and knows where she is in the task.	Hard to find the button.
Action 12: **Tell the system to remove all course events.**	Yes, the user knows the action, as it is common.	Yes, the user finds out the correct component, because it is clear.	Yes, the user finds there is no course event.	Yes, the user stays at the front page and knows it is the end of the task.	No comment.

Table 1: Action sequence and questions for Cognitive Walkthrough.

1.5: Summarizing results

This MFP has several strengths. First, it implements lots of features that are strongly needed for a university student. Then, the overall structure and layout are clear and colourful, which makes the user feel comfortable. Also, there are different kinds of interaction so that the user does not feel boring, and users understand most interfaces.

But the MFP also has some problems. The main problem raised by users is that some buttons and dialogues/input boxes are not clear enough, and they are not sure whether they can click or interact in the edit area. In Action 4, the user indicated that she did not know that the dialogue box was interactive. The user's suggestion for modification is to put the words Event Title/Time/Location/Type outside the dialogue box/input box. In Action 9, a similar question occurs, that button seems not clickable.



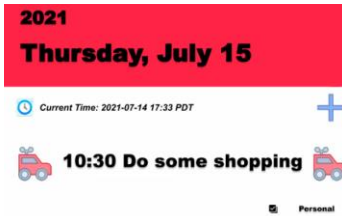
Event Title Default: Untitled Event

Event Time FROMAT: yyyy-mm-dd hh:mm

Event Location Default: TBD

Event Type Default: Personal

Figure 1: Action 4 - Screenshot of the edit area.



2021

Thursday, July 15

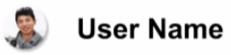
Current Time: 2021-07-14 17:33 PDT

10:30 Do some shopping

Personal

Figure 2: Action 9 - Screenshot of the pop-up window of July 15th.

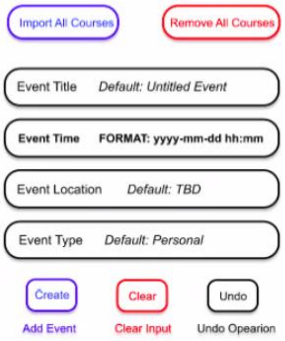
Then, the user also proposed amendments, that is, the "User Name" of the demo page can be changed to the name of a real person, like Joe, so that the user can have a sense of substitution during the trial.



User Name

Figure 3: Screenshot of the user information.

Finally, when the user concluded, she pointed out that in our overall UI design, the dialogue boxes and buttons on the right-hand side of the home page are too crowded. The user suggests designing a separation box to separate buttons with different functions. It will not only make the UI design more beautiful but also allow users to have a clearer view of the functions of the button group.



Event Title Default: Untitled Event

Event Time FORMAT: yyyy-mm-dd hh:mm

Event Location Default: TBD

Event Type Default: Personal

Figure 4: Screenshot of the right-hand side of the home page.

Component 2: Reflection

2.1: Design for the specific application interface

To provide suggestions on how to design an online calendar that facilitates different kinds of course activities for university students, we will offer the following lessons.

Lesson 1:

The first thing is to *identify and understand the context and user* of the product. The design group then could make surveys or interviews with randomly selected students from different universities, so that the group can learn about their thoughts about calendars, their habits or requirements of using calendars, and under what circumstances they would use calendars, etc.

At the same time, find some similar calendar products, like Google Calendar, and use *heuristic evaluation* to evaluate them. Identify these products' usability problems and good usability parts. These can make designers better familiar with the context of the user.

Lesson 2:

The second thing is to *gather requirements* from university students. Features of software and products have to be designed based on target users' demands and requirements. Especially, designers need to find out what are some of the most important features of a university calendar. When users' demands are met by these features provided by the calendar, they will be attracted easily. The requirements gathering will attract more users, thus improving products' popularity among students.

Through the previous lesson, the group should have raw data of student's views of the calendar. Then the group needs to analyze the requirements put forward by these volunteers, and figure out functional and non-functional requirements of the product. For example, the group can count the number of requirements that are mentioned, categorize these requirements by functional and non-functional ones, then choose the top three ones in each category. Don't have too many requirements for the product!

Combining Lesson 1 and 2, the group should write a clear and correct *specification* document pointing out the context and user identification of the product, three functional and three non-functional requirements for further steps.

Lesson 3:

The third lesson is to *design the prototype* according to the requirements. Don't design an MFP or HFP initially, which is a waste of time. The group should start by trying to design different LFPs, brainstorm with the team, and try different UI frameworks. It is important to realize these requirements in different LFPs, compare these LFPs, pick out their respective advantages, integrate them into a new LFP and compare them again. Finally, find a satisfactory LFP design into MFP. The group can invite some students to participate in the design, as the design should be user-friendly.

Lesson 4:

When the group has interactive MFPs, the group can *make a short presentation* to introduce the MFPs to the guests and companies and collect their suggestions.

Then the group can perform product evaluation on the designed MFP. By looking for an expert to predict user behaviour, let the expert help in the evaluation. Inform the expert about the context and scenario of the product to be used. After that, let the experts use heuristic evaluation to evaluate the MFP so that the group can get more general design and usability issues. The group can also use V-MFP to cooperate with experts to complete one or more *cognitive walkthroughs*. This evaluation method can obtain more detailed problems. It is recommended to use both evaluation methods so that you can get more comprehensive feedback.

Lesson 5:

The last is to *keep UCD in mind*, no matter in which step of the design process. Even after evaluation, the group should return to the first step of the design process, until the group has the final product.

2.2: Tools/methods used and how they help you to create a better design

2.2.1: Methods

Method 1: Heuristic Evaluation

We realize that the heuristic evaluation can help designers logistically check usability during the process, as it provides a standard evaluation method for designers, so we can find a way to design our works better. When conducting the heuristic evaluation of some common calendars, we find out the good usability and usability problems of the calendars. And then, when designing our project, we can avoid the problems and adapt good usability in our design to improve the quality of our product.

Method 2: Requirements Gathering & Specification

Gathering requirements and writing specifications are important when usability is created and designed. Only when we have clear and correct context identification, user identification, functional and non-functional requirements, we can continue the design smoothly. To collect the information, we should first ask ourselves questions about when/where/whom/what/how to use the calendar and figure out at least two personas of the product. We find functional and non-functional requirements, through which our product can better meet the requirements of users. We do not need too many requirements, but we should keep each requirement clear and detailed.

Method 3: Prototyping

We find the importance to create prototypes since they represent the core functional features. They also help designers check and evaluate the feasibility of their designs, so we can evaluate the details of designs and propose feasible solutions to potential problems. And we move forward to the final product by making modifications.

Based on the requirements, everyone should provide one LFP using Balsamiq with a detailed description so that we can easily choose or combine these LFPs. From LFPs, we design MFPs using Figma. But designing an MFP from LFPs is a major development in prototyping. Because a qualified MFP means the basic UI and functions of a product have been finalized. MFP is divided into H-MFP and V-MFP.

When designing H-MFP, we need to realize some main functions instead of all functions. H-MFP provides an essential product concept in the early overview. And when designing, we consider the logical relationship and user experience of these UIs for a better design. For example, we consider what it does when a button is pressed. When the H-MFP is done, the work of V-MFP starts. Otherwise, small modifications of the H-MFP may trigger lots of changes in the V-MFP. When designing V-MFP, we implement nearly all features of a meaningful and sophisticated task. To create a user-friendly design, we consider the user’s action in this task, design interactive components and animation of the V-MFP. To create a robust design, we let the product handle some issues like error typing. Through our design, we should let users understand what is happening and associate the feedback of the interface with the action to improve the design. Prototypes are cheaper, faster, and easier to build than the real thing. MFPs are similar to the final product, but we need evaluations.

Method 4: Cognitive Walkthrough

Through analyzing cognition, we can improve our design, and even create more interesting and amazing features for the final product. But we should first figure out context, scenario, target users, and then design a representative task with several actions. We invite a UX expert to do the CW following each action and pay attention to the feedback and comments since these are from a user’s perspective. We find some drawbacks of the current design in detail, and do some improvements like rearranging events and some UI components. The design is close to the final product now.

Method 5: UCD

Familiarity with the UCD process is vital, as the design is user-driven. We identify the target user, clarify the user's requirements, solve the user's needs, invite users to evaluate, and then loop again until we have the *final product*. We focus on the user's requirements and experience, and we let the product be user-friendly instead of forcing users to adapt, which is the design philosophy of UCD.

2.2.2: Tools

Tool 1: Balsamiq

Balsamiq is for designing LFP. It has a toolbar with different components, making the design easier. It is convenient for designers to draw, add marks and comments. Although it does not support team cooperation, Balsamiq simplifies our MFP design.

Tool 2: Figma

Figma is a for designing MFP. Figma has many functions and is easy to use, especially it allows us to import pictures and use some plugins. The component functions are easy to use and provide various interactive methods. The best part is team cooperation where designers can communicate and invite others. Although we may have to draw some frames on our own, Figma makes our MFP design simple.