Quality Management Plan

Team 2

2014210011 고나현 rhskgus@korea.ac.kr
2013210086 김영훈 2013210086@korea.ac.kr
2014210101 김원경 kwk2392@korea.ac.kr
2013210081 박진호 onlylove6913@kroea.ac.kr
2013210039 이민섭 alexn0@korea.ac.kr
2014210065 이상진 eesj@korea.ac.kr
2015410005 이정섭 james319@korea.ac.kr
2013210020 임원준 rhimjun@korea.ac.kr
2014210059 정희석 poco2889@korea.ac.kr

Table of Contents

Ⅰ. Understandability

Definition : How easily a user (a programmer and a customer both) understands the function of a program.

■ How much is good enough?

① The capability of a programmer: Each programmer is required to maintain or develop the program with his/her abundant experience and deep understanding of a programming language.

② The architecture of a program: A programmer should be able to fix the problem when errors occur. For this, a source code needs to be simple so that other programmers can easily understand the whole structure to give feedback or revise.

③ The application design (UI/UX): When a customer searches a photographer or a photo, he/she can understand easily how to use. The application should be designed explicitly in order for a customer not to get lost.

■ How can you make quality plan?

① The capability of a programmer:

The way of development that each programmer uses should be synchronized through pair-programming. By pair-programming, a programmer can improve his/her capability to implement the program.

A programmer should join the follow-up meeting with other members so that he/she enhances the understandability of parts he/she does not know. It will eventually improve the performance of a program.

② The architecture of a program:

The name of a variable or a function should be recognized intuitively. If a variable has the same meaning or same type of a value, each name of a variable should be called consistently in the whole structure. In this manner, a name of each function should be also described identically if it performs in the same way.

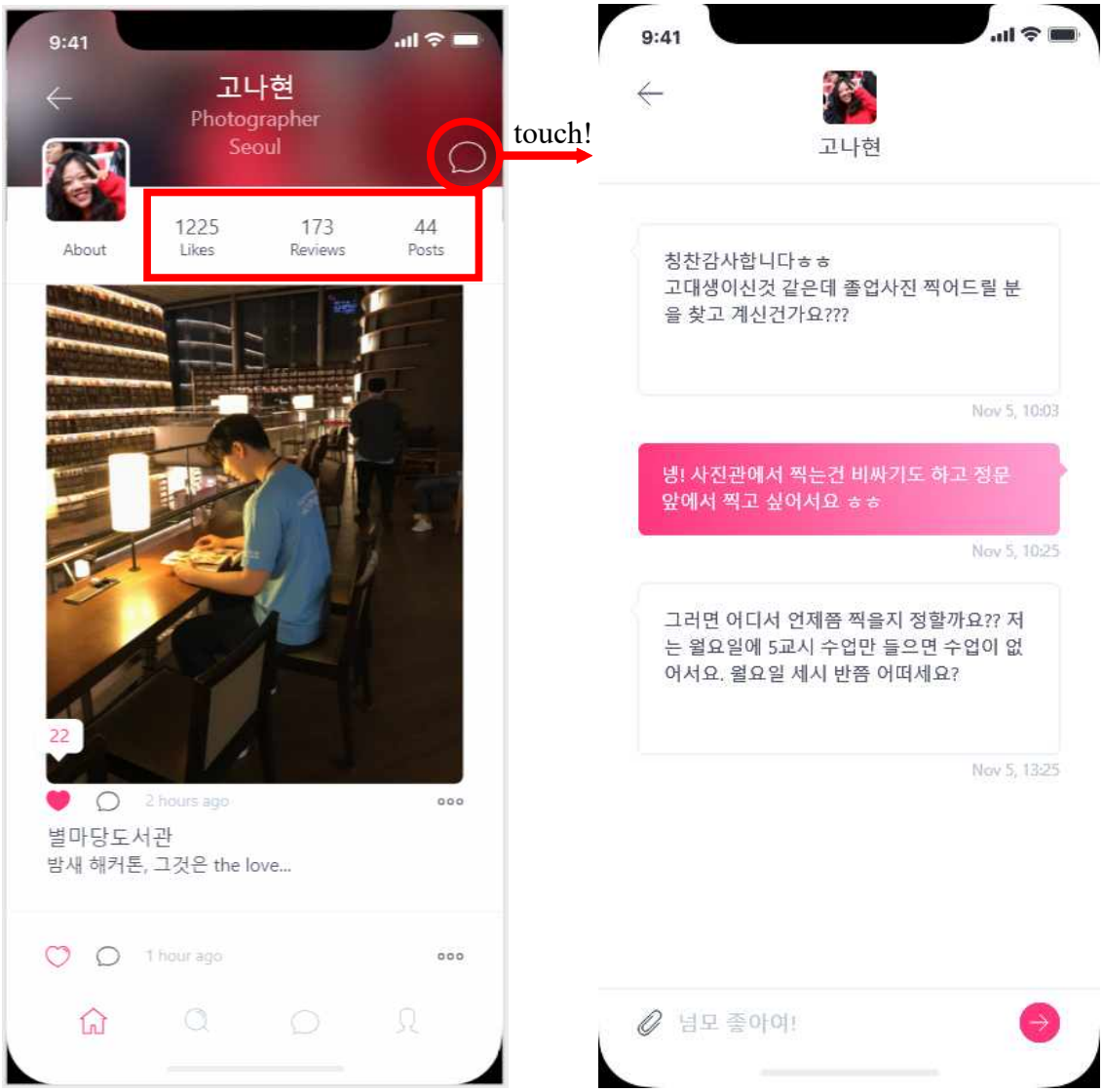
Frequently-used functions should be modularized to make the code consistent and main the architecture.

③ The application design (UI/UX):

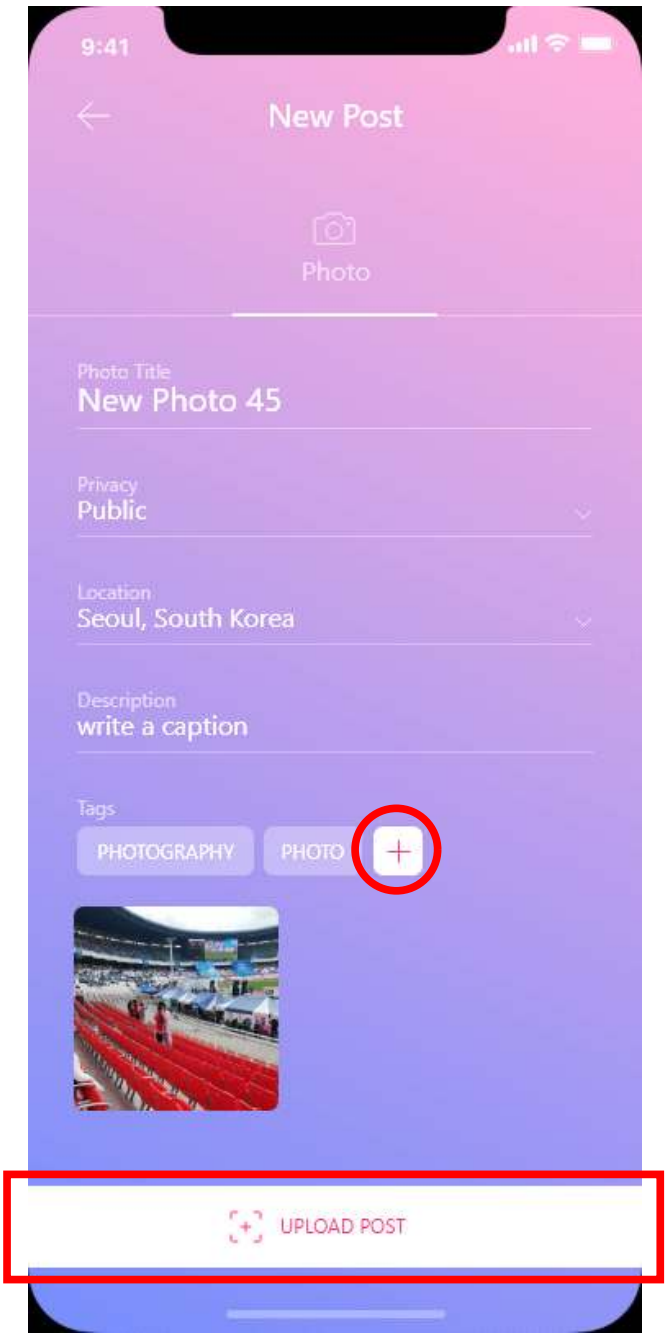
Buttons of main functions such as exchanging message, giving “Like”, uploading photos, managing a profile are designed to be simple and explicit. Users can recognize each button intuitively.

Each menu will be displayed in the proper place to prevent an user get confused when using the application. If all menus are gathered in one spot it will results in increasing the complexity.

A simple tutorial will be offered in the beginning to maximize a customer’s understanding.



: The button of ‘exchanging message’ is described in this way. An user easily catches the concept of the button intuitively. The number of “Like”, reviews, posts is separately written in the upper of the page; an user understands them without confusion.



: Each menu/buttons in the page should be well-organized so that an user finds them easily.

Ⅱ. Usability

■ How much is good enough?

① It shows how change in the page comes out by manipulation so that users do not lose their context. Sudden changes make users confused.

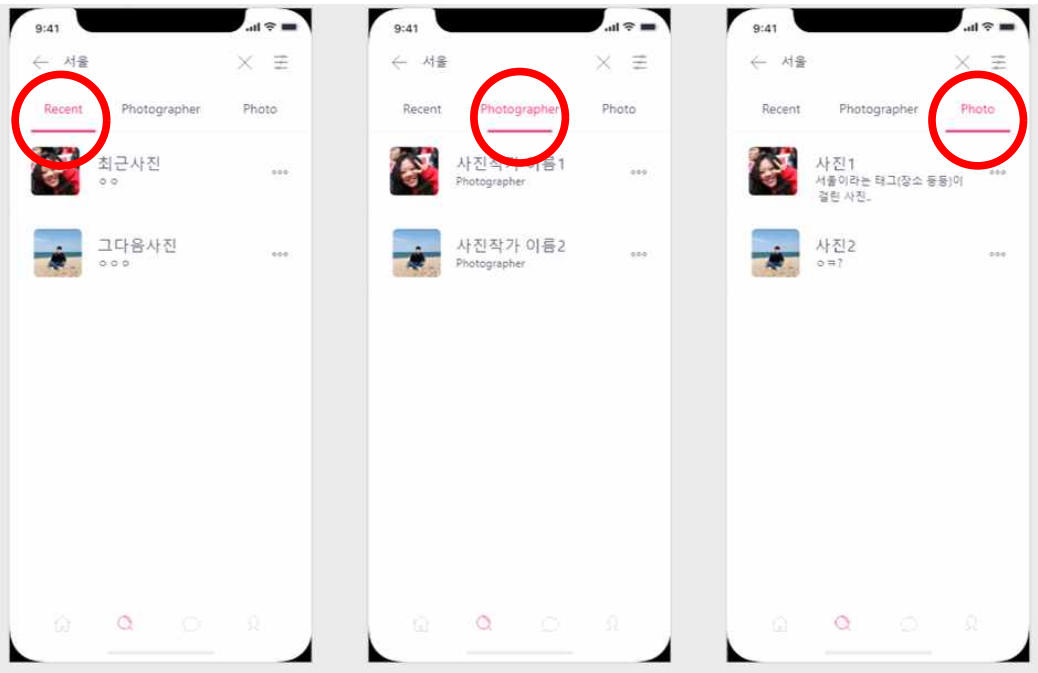
② By viewing the results simultaneously with real-time operation, users can save unnecessary time and lessen the number of actions.

③ It is a bad design, which has too many points to see, to use the platform. To avoid a bad design, it provides brief information first and leads to the next step which contains more detailed information.

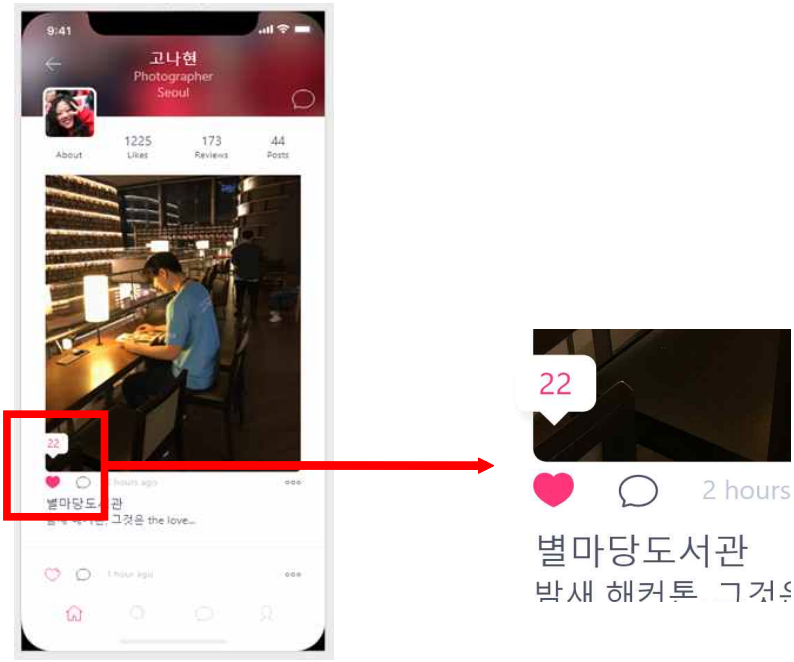
④ It provides simple examples how the platform performs.

■ How can you make quality plan?

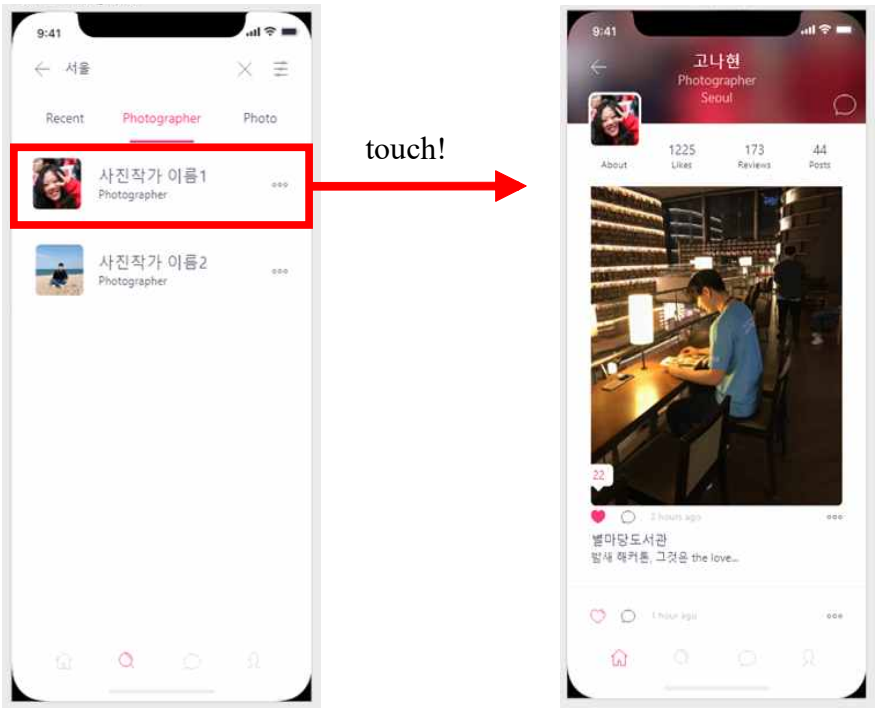
① The search page shows which category the user is currently in.



② Pressing the "Like" button will instantly show how many the total number of “Like” goes up.



③ It provides simple profile information and gives choices to make decision whether to see more information or not.



④ Attach a short videoclip that shows how platform operates.

Ⅲ. Reliability

■ How much is good enough?

$MTTF$ is Mean Time to Failure
$MTTR$ is Mean Time to Repair
$MTBF = MTTF + MTTR$

$$A = \frac{MTTF}{MTBF} = 1 - \frac{1}{MTTF}$$

- Decreasing to 0.0001 (which $MTTF$ is 10,000 hours)
- Decreasing $MTTR$ to 24 hours
- Increasing $MTBF$ to 10,024 hours
- Make A to 0.9976 $\left(A = \frac{10{,}000}{10{,}024}\right)$

■ How can you make quality plan?

- Build 2 Servers
- Put as many team members as possible to get the fastest recovery.