**Abstract**

With highly developed information technology, people have become accustomed to using virtual keyboards to type and text in various situations. In many scenarios, people need to type with only one hand and use the other hand to do other things. Nonetheless, one-handed interaction with mobile devices is influenced by multiple factors, including variations in keyboard functionality. Notably, mobile device keyboards still lack comprehensive support for one-handed typing, potentially leading to accidental touch, missed touch and other behaviors that can hinder typing efficiency. Our goal is to better understand the user experience of typing with only one hand, and to explore ways to improve the effectiveness and efficiency of typing on mobile devices with one hand. In this article, we provide a detailed description of our study along with our initial design for one-handed typing. In the past weeks, we employed a detailed survey and conducted contextual inquiries to study the potential difficulties and motivation of one-handed typing on mobile devices and developed 6 user requirements. We then established a paper prototype that contains key features such as adjusted key width, improved autocorrect, and revised punctuation keyboard. In this assignment we evaluated our paper prototype via simplified user testings and heuristic evaluation, and evaluate the results against our user requirements. The results show that our design fails to reduce recognition load and might have visibility problems in system status. We will use these feedback to build the high-fidelity prototype in the future.

**Introduction**

(The previous are the same as assignment 3)

After building our paper prototype, we moved to the interactive systems evaluation section. We firstly revised the wording of user requirements to improve its testability and objectivity. Then we separately conducted heuristic evaluation with usability experts to identify usability issues with respect to the 10 usability heuristics. Our results show that we still have usability problems mainly in visibility of system status, user control and freedom, and recognition load. We also conducted simplified user testing with stakeholders to test the features and usability of paper prototype against our user requirements. In this task we encouraged the participants to ‘think aloud’ while doing the tasks and we acted as observers to watch for errors and confusions. Our results show that the participants still make errors and get confused when completing several subtasks. In short, we evaluated our paper prototype using simplified user testing and heuristics evaluation to get valuable feedbacks to improve the existing prototype.

# 

# Final Primary Persona

# Final Sketches & Storyboards

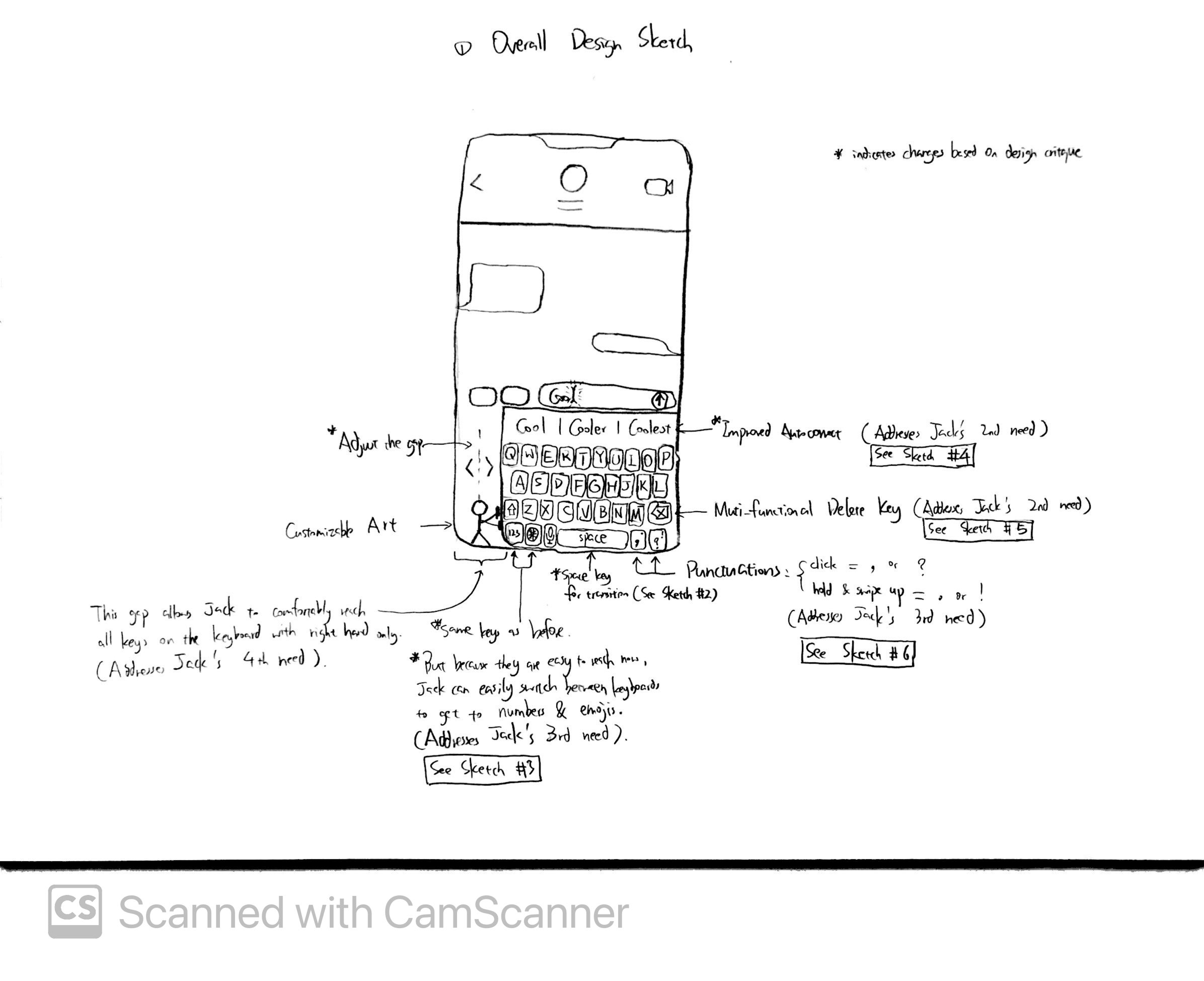
## Final Design

The final design (sketches & storyboards) were created based on the feedback received from design critique. Each team member actually had very different designs, meaning that our user requirements were well-written as they do not imply a specific solution. First, we tried our best to merge the different features of our designs that attempted to resolve the same need of the persona (i.e., addresses the same user requirement(s)), but if we struggled to do so, we collectively voted to keep the best one and discard the others. For example, some of Jerry’s, Yichen’s, Daniel’s, Franklin’s features all addressed the same need of being able to reach all elements on the keyboard (Jack’s 4th need, User Requirements 4, 6), and they were vastly different designs. After careful consideration, we decided to keep the one that is the most feasible (for the paper-prototype and wizard-of-oz) and adequately (in our opinion) resolves the persona’s needs. Next, for each of the features we decided to keep in the final design, we redrew/modified the sketches and also tried to improve on them based on the critique (see each of the following sketches to see the things we improved based on the design critique). Lastly, the storyboards were modified to illustrate the final primary persona and the context of use for our final design.

## Final Sketches

Sketch 1 presents the new one-handed keyboard as a whole, which attempts to address all of Jack’s needs. Sketch 2 shows the process of transitioning between the existing two-handed keyboard (current design) and the new one-handed keyboard (my design). Each of the remaining sketches (3-6) shows a specific part/feature of the overall keyboard, explaining how it attempts to address one (or more) of Jack’s needs.

### Sketch 1



This sketch illustrates the overall one-handed keyboard layout and general description of its features/keys. One major difference compared to the two-handed keyboard is the shift of the keyboard to the right side of the screen. By doing so, Jack should be able to easily reach all keys, including those that are on the left edge of the keyboard, thus accounting for Jack’s 4th need, where he would like to comfortably reach all elements of the keyboard while holding his phone securely. I did not make any significant changes to the layout of the letters because Jack is already so familiar with typing in the “QWERTY” layout, consequently, switching to a drastically different layout not only takes a long time to learn, but also may not even make any meaningful difference in meeting the user requirements. Thus, I just went with making gradual improvements to the existing layout instead of coming up with a breakthrough. One additional feature we added to this was the ability to adjust how much the keyboard is shifted or squished towards the right (see Sketch 3 for more details).

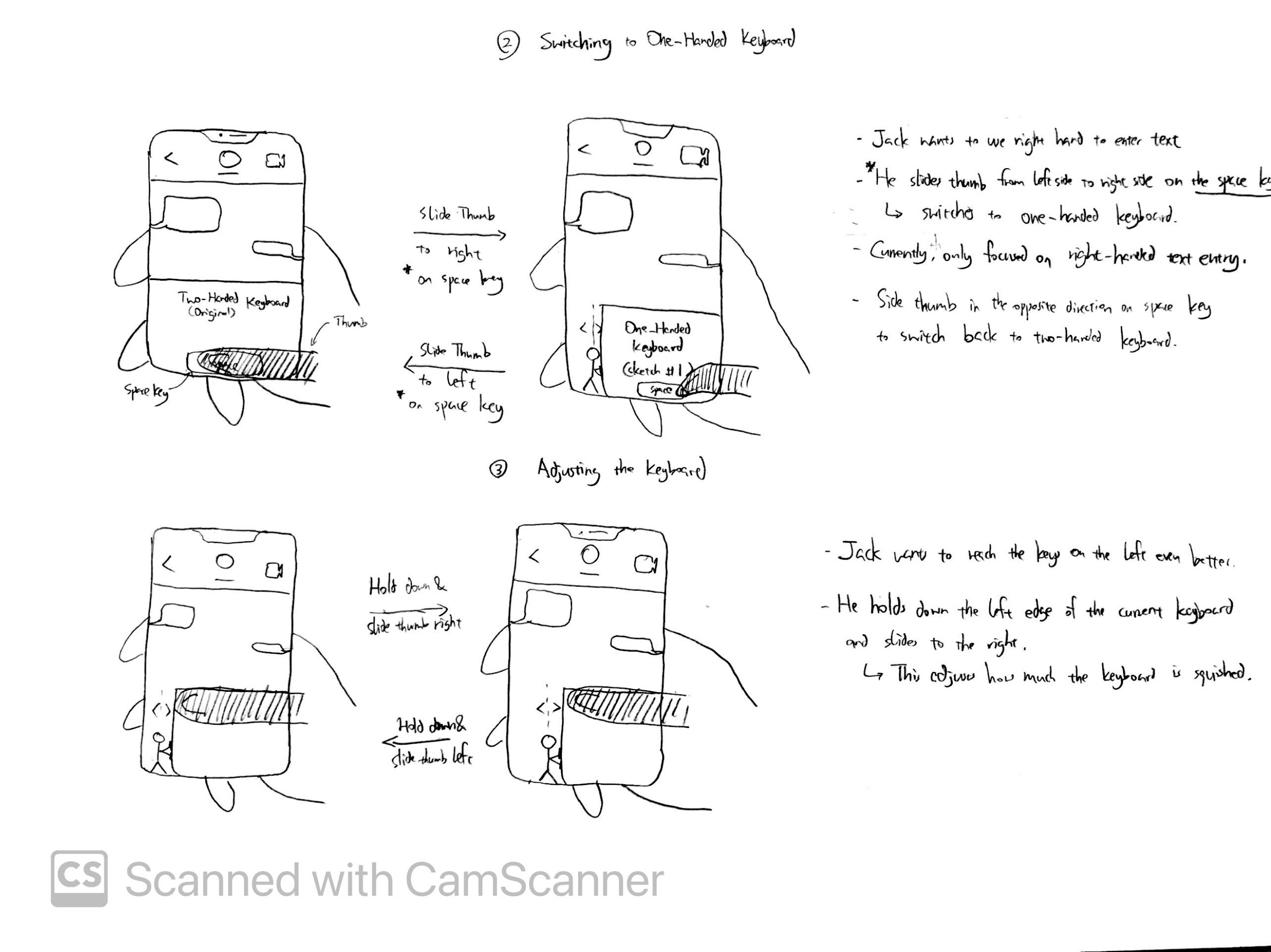
Another noticeable difference is that the “return” key at the bottom right corner is replaced with two separate keys. Before discussing what those keys are, I would like to elaborate on why the return key is removed. Jack is usually in informal situations when using his phone with one hand, such as texting, Google searching, or browsing social media. This means that the “return” key does not function as creating a “newline”; instead, it serves the purpose of “send” (sending his message, his search query, or his comment). However, these apps which Jack uses to text, search, or browse social media, already have a separate “send” key apart from the keyboard, so the “return” key in these situations are actually redundant. Therefore, it is removed and I believe that it does not cause any reduction in functionality. As for the two new keys, they attempt to address Jack’s 3rd need which is to conveniently enter certain characters; specifically, these two keys are for punctuations (see Sketch 6 for details).

Originally, this keyboard also had 2 additional special keys that attempted to address the remainder of Jack’s 3rd need, which was to easily enter *numbers* and *emojis*. However, based on the design critique, I realized that these keys not only would make the keyboard’s bottom row too squished, but they were also a little redundant since the fact that the keyboard being shifted towards the right already implicitly addressed the need to easily enter numbers and emojis, because Jack would now be able to easily reach the emoji/number-switch key on the left side of the keyboard, which was the main breakdown we identified from our contextual inquiry results. Thus, these two keys were removed in the final design.

Despite not having any salient differences in terms of their appearances in this sketch, the autocorrect feature (see Sketch 4) and delete key (see Sketch 5) both have improved functionality, in order to resolve Jack’s 2nd need, which is to easily fix any typos.

Lastly, although Jack’s 1st need is not explicitly addressed by any particular key/feature, I believe that when considering all of the features together as a whole, this new one-handed keyboard would be able to reduce the frequency of typos, thus resolving Jack’s 1st need.

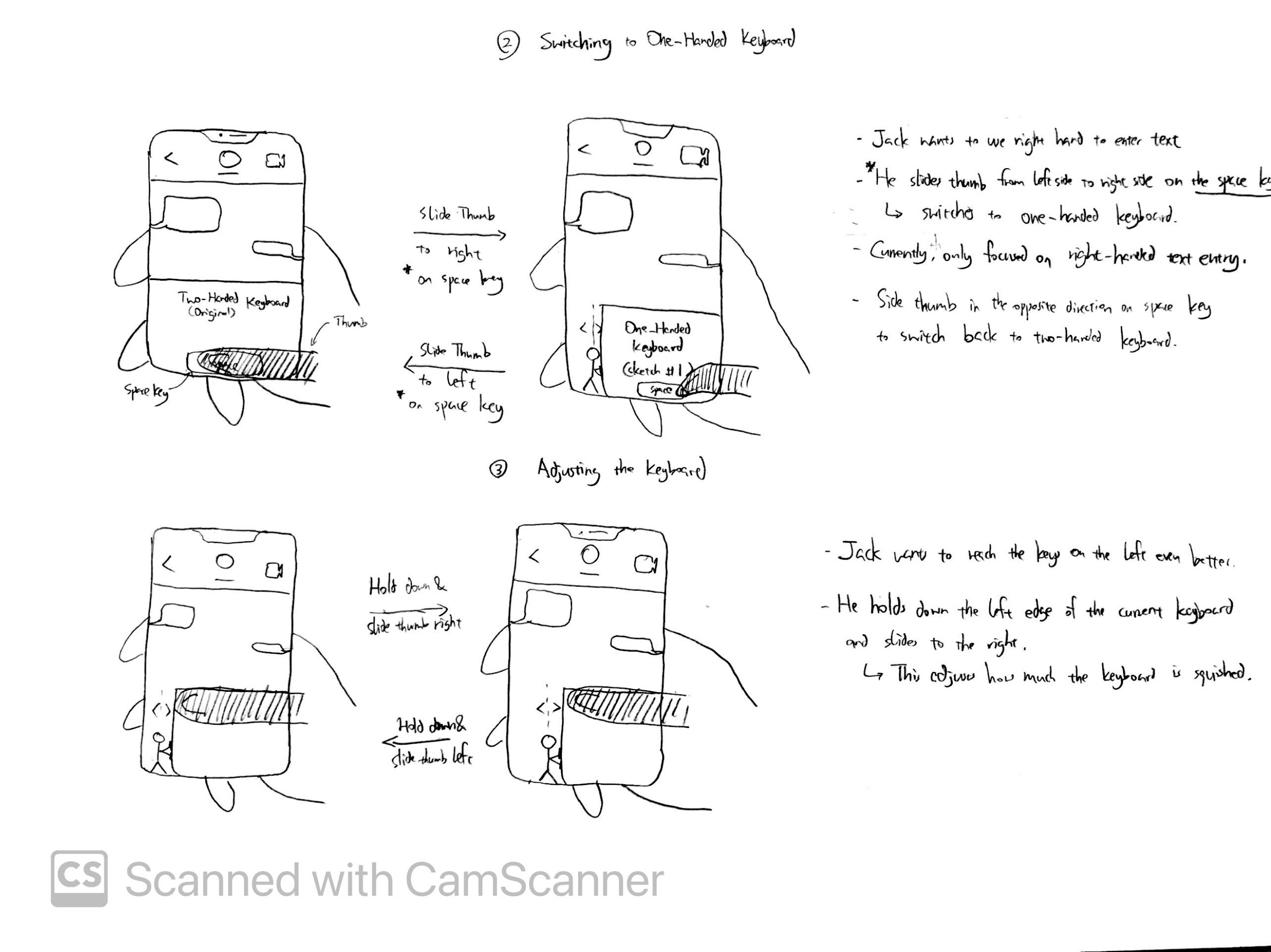
### Sketch 2



This sketch shows the process of switching from the existing two-handed keyboard to the new one-handed keyboard (my design), and vice versa. To do so, Jack simply needs to slide his (right) thumb from the left side on the keyboard’s space key to the right side (as shown in the sketch). To switch back, Jack simply needs to swipe in the reverse direction (from right to left). Note that, since Jack is right-handed, the one-handed keyboard and this transition motion are designed for people who type with the right hand (thumb). In the future, we plan to design for people who mainly type with their left hand as well.

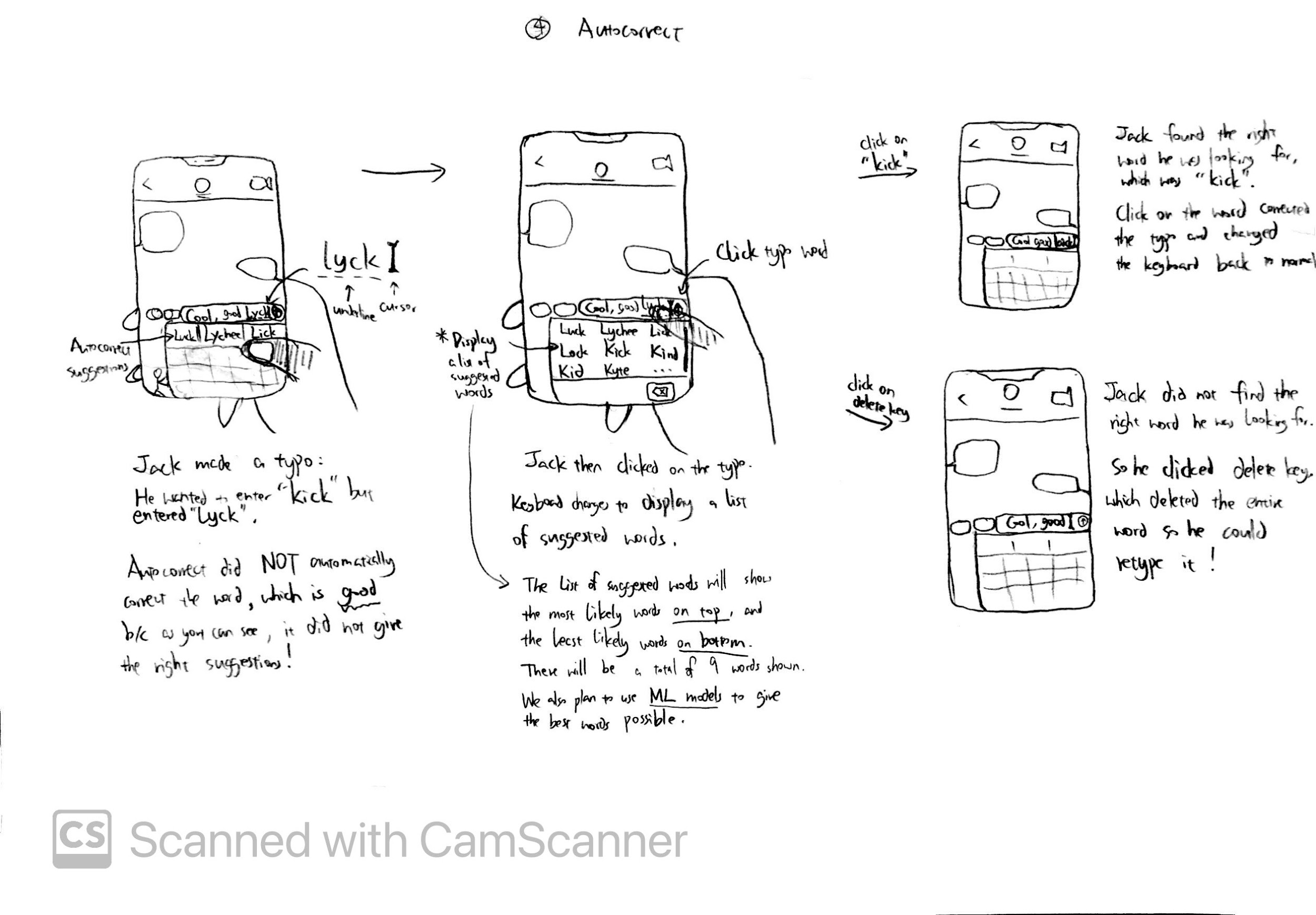
This feature was improved based on the design critique. Originally, this feature allowed the user to swipe anywhere on the keyboard to switch between two-handed keyboard and one-handed keyboard. However, during the design critique, we pointed out that although this allowed a convenient transition, swiping anywhere on the keyboard might interfere with other features on the keyboard and the screen/app, for instance, the swipe-to-text feature and the “back” feature (which are existing features on the iPhone). Thus, we decided to change the area which can pick up this interaction to just the space key, which does not have any existing features when the user swipes on it.

### Sketch 3



This was a newly added feature based on the design critique, which allows Jack to easily adjust the width of the new keyboard, or how much the keyboard is “squished”. This gives him the flexibility since Jack might be holding the phone with different hand positions from time to time, so the need of how well he needs to reach all keys on the keyboard may change. Thus, the addition of this feature addresses Jack’s 4th need more optimally.

### Sketch 4



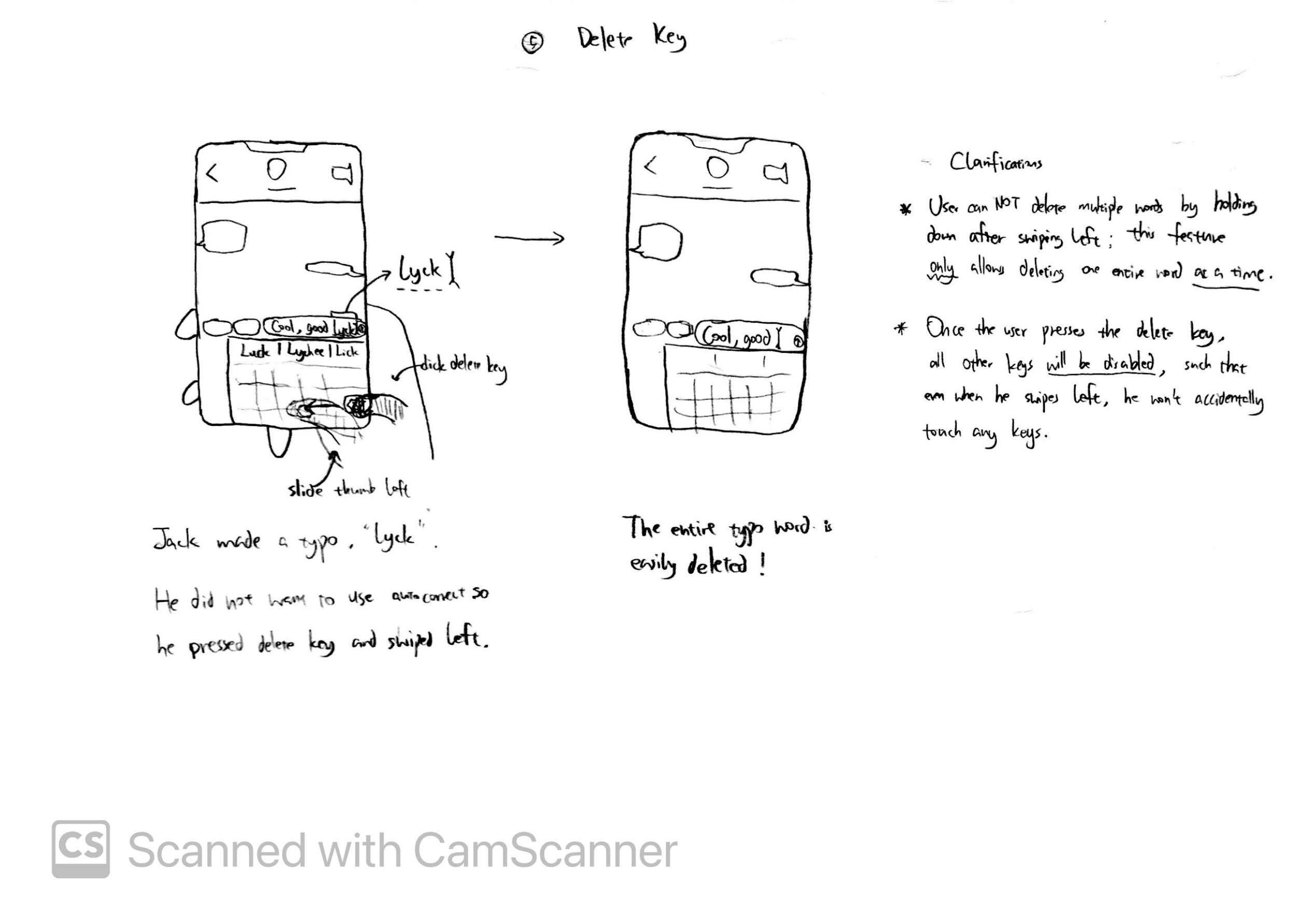
The autocorrect feature in the original two-handed keyboard can be counterproductive, specifically when it automatically changes a typo to a correctly-spelled word but is not what Jack intended, which happens frequently. In order for autocorrect to not automatically change the typo, Jack has to manually select one of the three suggested words displayed at the top of the keyboard, which he tends to forget especially when he is multitasking or distracted, resulting in autocorrect being counterproductive when Jack is entering text with one hand.

In our new one-handed keyboard, autocorrect no longer automatically changes the typo; it only underlines the typo and Jack can choose to manually change it or not. If Jack would like to change the typo, he simply clicks on the word and autocorrect will display a list of 9 words on the keyboard for him to select from; he can also scroll down the list of words to see more suggestions. If he finds the right word, he simply clicks on the word and Autocorrect will replace the typo with the word clicked. If cannot easily find the right word, he simply clicks on the “delete” key which deletes the entire typo, and he can then just retype it.

Based on the design critique, we also decided to incorporate Machine Learning models that learn from the patterns which Jack types to produce a better list of words that have a really high chance of giving the right corrections (merging with another feature by a different team member). We also changed the order in which the words appear to display the most likely words (the algorithm believed to be correct) at the top, and the less likely words at the bottom. The list would also dynamically update based on Machine Learning.

This improved functionality takes care of Jack’s 2nd need (specifically part a), because he no longer needs to fix any mistakes made by autocorrect, therefore, the autocorrect feature is not counterproductive.

### Sketch 5

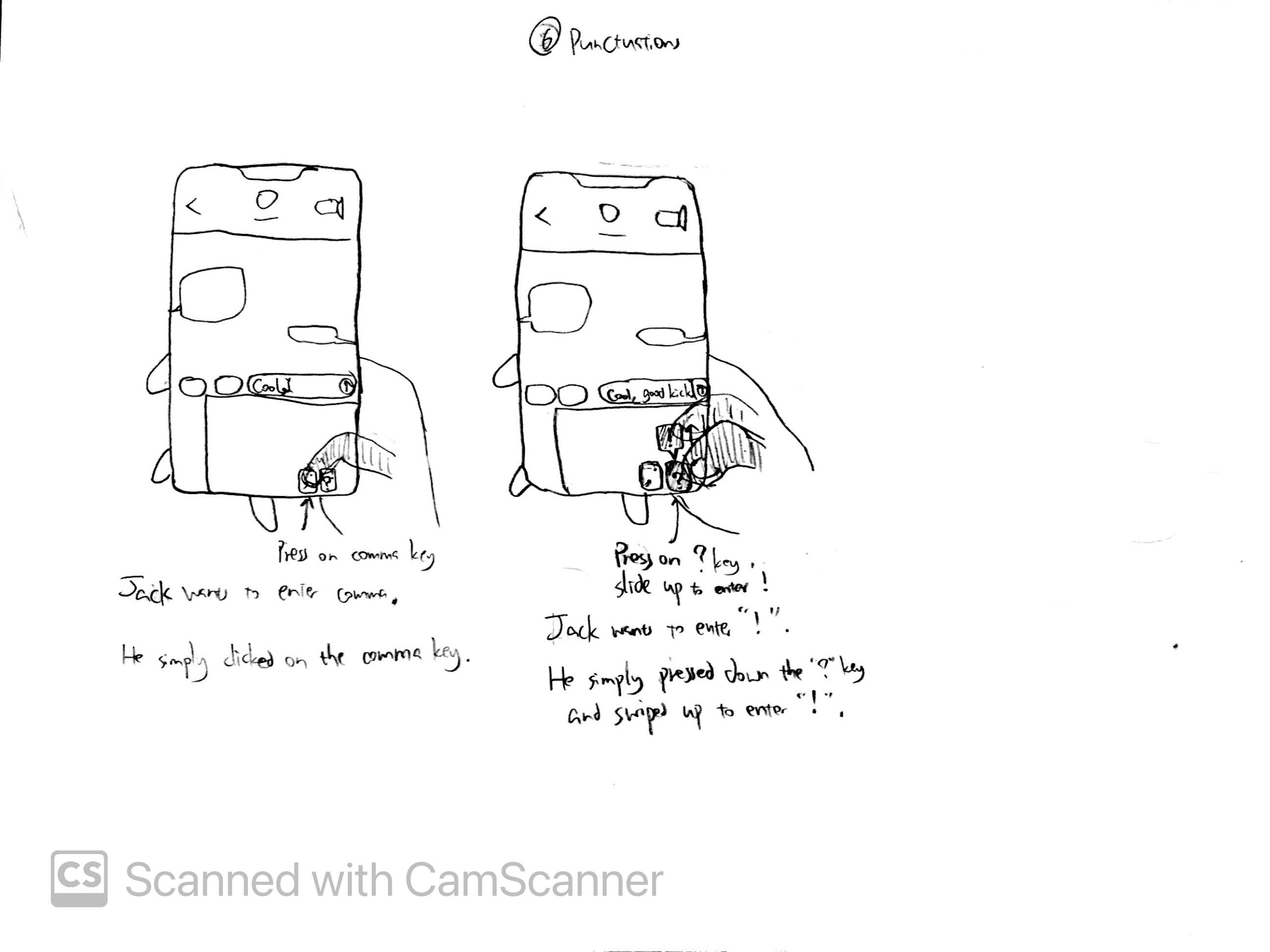


In the original two-handed keyboard, holding down the “delete” key will continuously delete one character at a time at a fast pace. This functionality is very difficult to control, as reflected in Jack often deleting not just the typo, but also the correctly-spelled word(s) in front of the typo. Thus, the delete key is originally counterproductive.

In the new one-handed keyboard, the “delete” key keeps the original functionalities, but also includes a new functionality: Pressing the delete key followed by sliding left will delete an entire word. This allows Jack to quickly and accurately delete the typo itself and nothing else, thus eliminating any counterproductivity caused by delete, resolving Jack’s 2nd need (part b).

This feature was also improved based on the design critique, which we realized that the swiping left interaction may result in the user accidentally clicking on the keys next to the delete key. Thus, to fix this, whenever the user clicks on the delete key, all of the remaining keys automatically are disabled (i.e., clicking them would not result in any text entry / functionality). A visual would also be shown that the keys cannot be clicked. As such, the delete key would accurately delete the words without causing more slips, better resolving Jack’s 2nd need.

### Sketch 6



In the original keyboard, whenever Jack has to enter a punctuation, including the frequently used ones like comma and question mark, he has to switch to the numbers/punctuations keyboard by pressing the keyboard-switch key at the bottom left, locate/enter the punctuation, switch back to the letters keyboard. He has to perform these three steps whenever he wants to enter any punctuation, which is complicated and difficult, especially when operating the phone with just one hand.

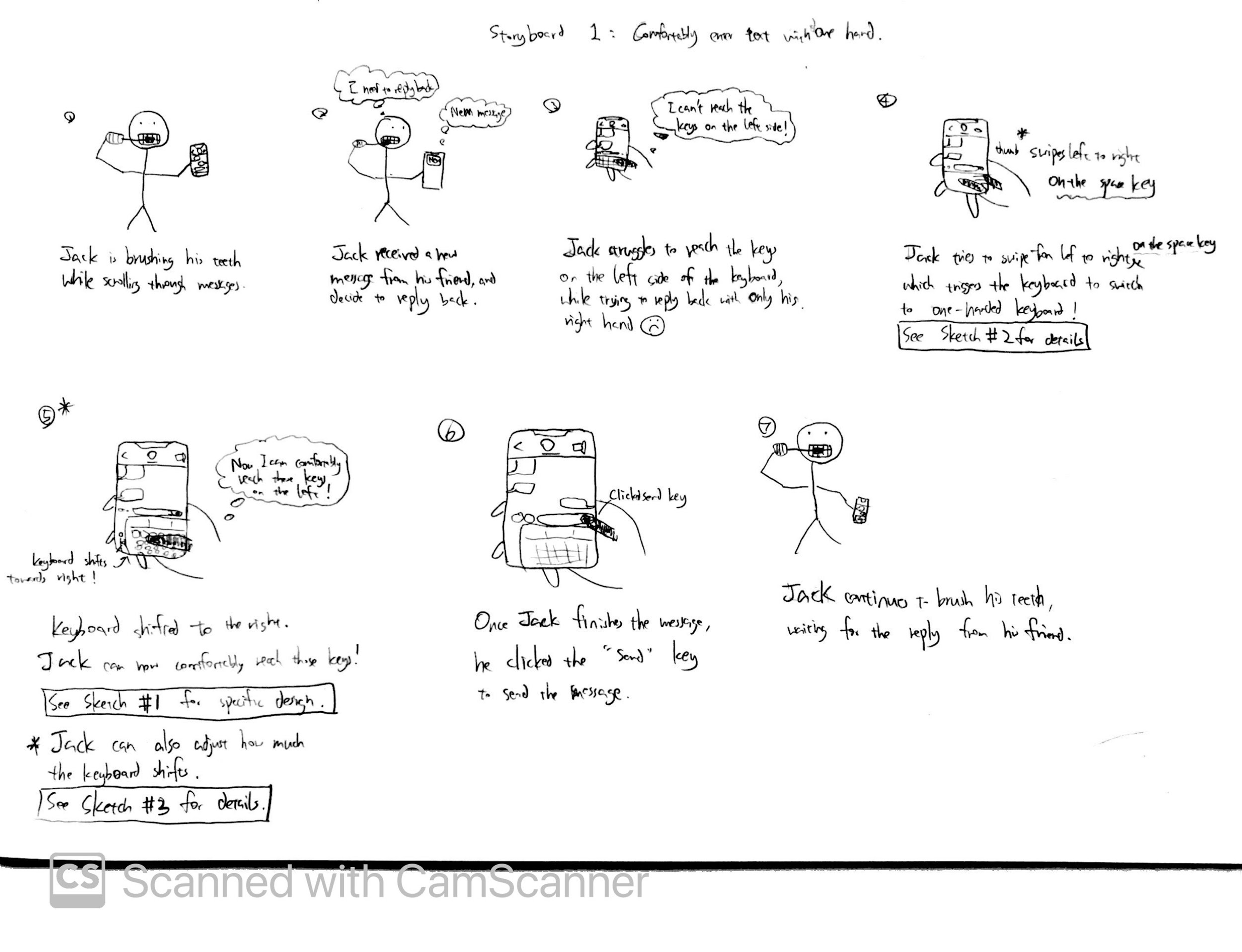
In the new one-handed keyboard, the punctuations key at the bottom right corner allows Jack to conveniently enter four frequently-used punctuation marks with just one step, which is pressing on the corresponding punctuation key, eliminating the excessive steps of keyboard switching. These two keys have default values of ‘,’ and ‘?’ respectively, which can be entered by a simple click. They also offer two alternative values of ‘.’ (period) and ‘!’ respectively, which can be entered by clicking and sliding up, which despite requiring one extra motion, is still much simpler than switching keyboards back and forth. Thus, this new addition of punctuation keys will account for part of Jack’s 3rd need, which is to enter non-alphabetical characters, in this case, the punctuations, conveniently.

Like mentioned previously under Sketch 1, this keyboard also had 2 additional special keys that attempted to address the remainder of Jack’s 3rd need, which was to easily enter numbers and emojis. But based on the design critique, we removed those two keys as they were redundant as the remainder of Jack’s 3rd need would be resolved by the shifted keyboard already (Sketch 1, 3). Thus, this was changed after the design critique for this feature.

## Final Storyboards

Storyboard 1 extends on Sketch 1, 2, 3. Storyboard 2 extends on Sketch 4, 5. Storyboard 3 extends on Sketch 6.

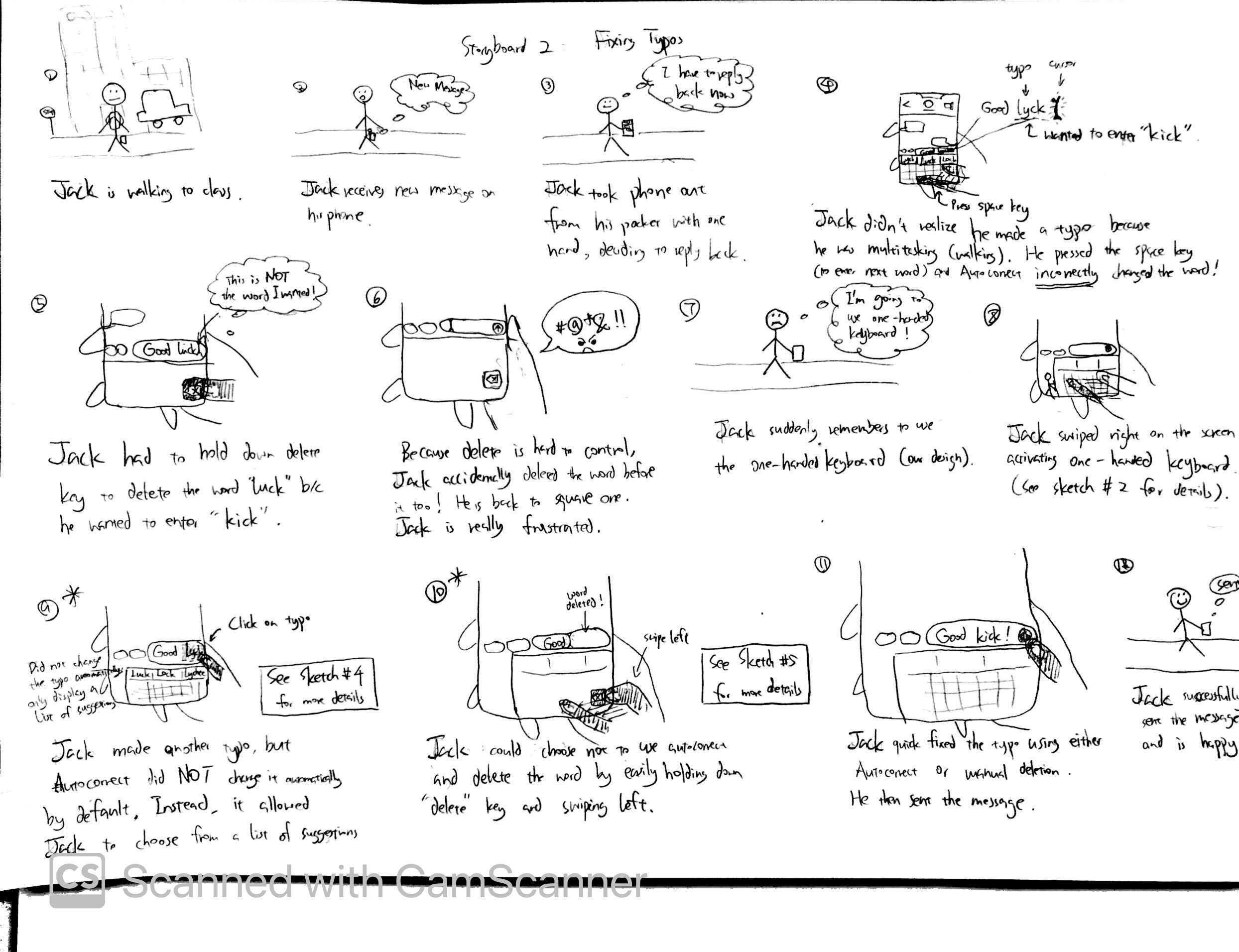
### Storyboard 1



This storyboard depicts how Jack, who is trying to reply to a message to his friend with one hand (specifically, his right hand) while brushing his teeth with his other hand, initially struggles with entering text on the current keyboard on his phone, because his thumb could not reach the keys on the left edge of the keyboard.

However, by changing (Sketch 2) to my new one-handed keyboard design (shown in Sketch 1 in detail), he is able to comfortably reach all keys on the keyboard without readjusting his hand position thus securely holding the phone. Consequently, he is able to effectively complete his message and send it to his friend. Thus, this successfully addresses Jack’s 4th need (and also part of Jack’s 3rd need).

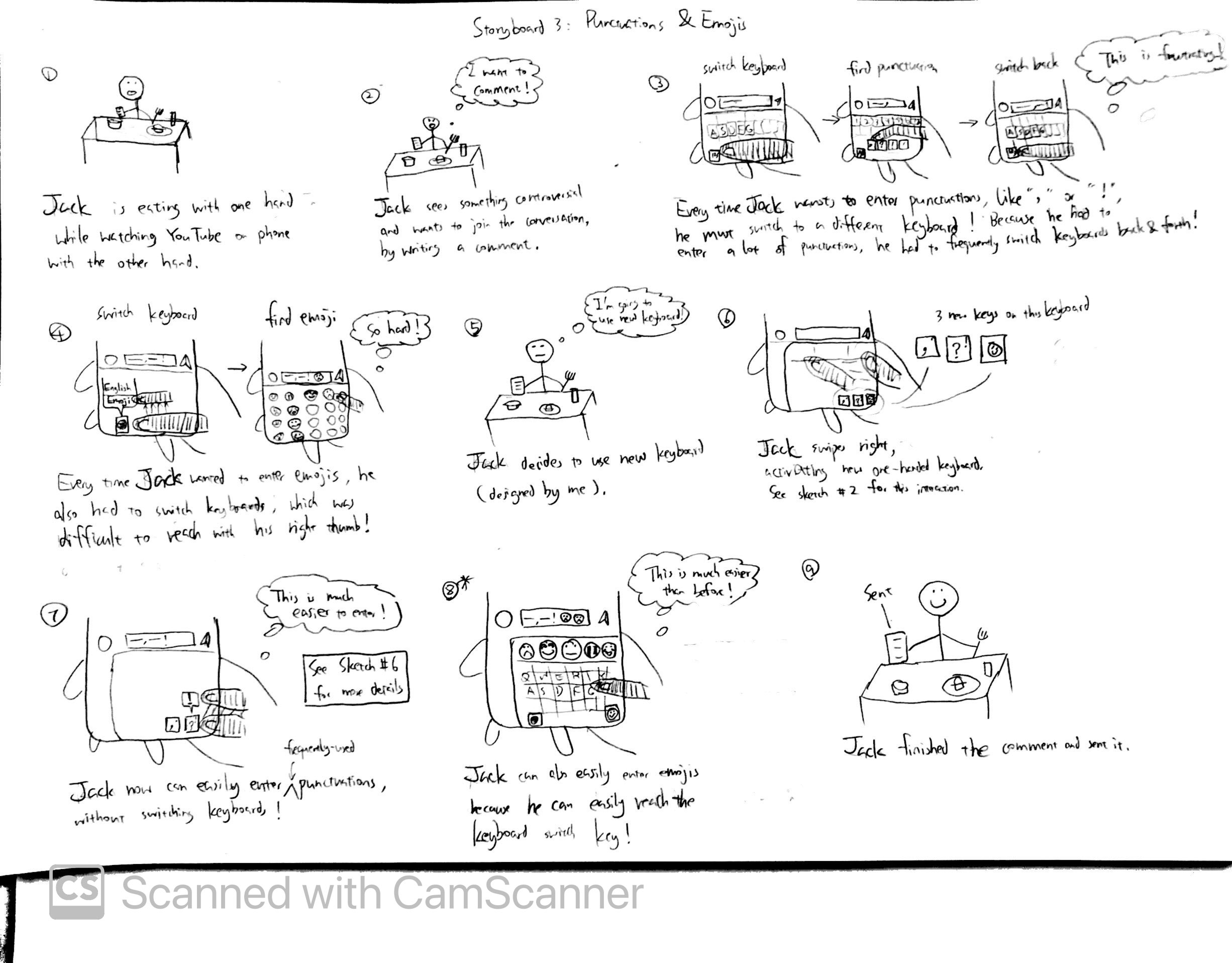
### Storyboard 2



This storyboard depicts how Jack, who is trying to reply to a message from his family member with one hand while walking to class, struggles to fix typos on the current keyboard design because the autocorrect incorrectly changes his typo to a word he did not intend to type, and while deleting this word, he accidentally over-deletes the previous correctly-spelled word too.

However, by changing to the new one-handed keyboard design, Jack is able to easily fix typos because the autocorrect and the delete key are no longer counterproductive, as the former allows Jack to manually pick from a listed of suggested word (Sketch 4), whereas the latter allows Jack to accurately and efficiently delete a word (Sketch 5). Consequently, he is able to efficiently finish the message and send it to his family member. Thus, this successfully addresses Jack’s 2nd need.

### Storyboard 3



This storyboard depicts how Jack, who is trying to comment on YouTube with one hand while eating food with the other hand, struggles to enter punctuations marks and emojis, because everytime he wants to enter one of them, he has to switch to a different keyboard, find the desired element, and switch back to the letters keyboard to enter the next word.

However, by changing to the new one-handed keyboard design, Jack is able to conveniently enter frequently-used punctuation marks, including comma, period, question mark, and exclamation mark (Sketch 6), and also easily enter desired emojis (and numbers) because he can now easily reach all elements on the keyboard. Consequently, he is able to efficiently finish the comment and publish it on YouTube. Thus, this successfully addresses part of Jack’s 3rd need.

# Final Paper Prototype

## Description

**Our final paper prototype was implemented closely following the features and interactions shown in the final sketches and storyboards (see Figure ? for the final paper prototype).** Because we attempted to satisfy all of Jack’s needs (primary persona) when creating our final sketches and storyboards, our paper prototype should also address those needs, and by extension, the six user requirements. To reiterate what we mentioned in the Final Sketches and Storyboards section, we have also recorded other feature design ideas that are not incorporated into this iteration of our prototype, mainly due to conflicting designs for a certain need/requirement. Our rationale for choosing one feature over the other was that the selected feature should adequately address Jack’s needs (and by extension, the user requirements) while being conservative with current keyboard layouts and being feasible for a Wizard-of-Oz. We did not yet want to incorporate features that drastically overhaul current widely-accepted solutions, but we may revise our prototype in the future and incorporate other features consisting of relatively more radical changes, given our user evaluation feedback.

The paper prototype consisted of 2 main parts: (1) frames and (2) flow between frames. Each frame, or a snapshot of the interface, in the prototype is built upon a sketch; it clearly shows the context for which the user is using the phone, namely, messaging a friend, and the keyboard design with its newly added features. The flow between frames, indicated by the red boxes and arrows, explains how/where the user should interact with the prototype to go from the current frame to the next frame, which is also built upon the sketches and the storyboards. For example, to go from the first frame, which depicts the current/existing two-handed keyboard, to the second frame, which depicts our new one-handed keyboard, the user should press the space key and slide right, as indicated by red boxes and arrows. All of the remaining frames and flows follow the same standardization, so it is easy to understand how the prototype works. It is redundant to go over each frame and flow of the prototype one-by-one (since these are similar to the sketches and storyboards), so we mainly focused on discussing the changes we made and how they were implemented in the final paper prototype (from sketches/storyboards) based on the design critique.

## Changes based on Design Critique

First, the design critique raised concerns that the activation of the one-hand keyboard may interfere with the default swipe-to-text feature of the keyboard if the user were to just swipe at an arbitrary location on the keyboard. To counter this issue, the activation for this feature was moved to swiping the spacebar instead, which served no purpose on the default (current) keyboard (see \*1 in Figure ?). Moreover, to increase the customizability of the one-handed keyboard, the user can now manually adjust how much they want to shift the keyboard depending on the magnitude of their swiping-action; the keyboard can now be squished until the characters symbols on the keys have zero padding (see \*2 in Figure ?).

Based on the design critique, the autocorrect feature is now utilizing a context-sensitive language model to make good predictions for misspelled words and the user is given a list to choose from (see \*3 in Figure ?).

The new delete feature is revised by deactivating all other keys on the keyboard while the user is pressing and swiping the delete key, so that the unwarranted pressing of other keys would not interfere with this action (see \*4 in Figure ?).

Lastly, having the emoji and number keyboard switch key at the bottom row of the keyboard was deprecated in the final paper prototype because this was deemed to be a redundant feature in the design critique. The original purpose was to help the user locate and enter numbers and emojis with ease; however, since the one-hand keyboard is already making edge elements easier to reach with one hand, the user could simply switch to emoji and numbers keyboard with the conventional way.

## Wizard of Oz

Because the nature of our paper prototype is broken down into frames and flow between frames (arrows), it can be easily adapted to a Wizard of Oz. To Wizard of Oz our prototype, we first need to print out each frame in the prototype on paper, where each frame should be about the size of an iPhone. We tell the participant in the simplified user testing that their goal is to send a message, “Cool, good kick!”, to their friend using the prototype with one hand, which is what our paper prototype is built around. The participant begins by physically holding the very first frame shown in Figure ? with one hand. Then, based on the participant’s interaction with the current frame, we can present different frames to the participant by quickly switching the current frame out from their hand with the next frame, according to the flow of frames, namely the arrows, shown in Figure ?. Because the participant is trying to do a specific goal, there are only a certain set of interactions (tasks) they could do with the keyboard, which would inevitably have them go through most if not all of the frames in the prototype. As such, we can use the paper prototype to conduct a Wizard of Oz that allows the participant to experience all of the features and functionalities of our design.

# Discussion

Regarding the prototype for this iteration, our group was able to come up with diverse design ideas for addressing our user requirements.

The user requirement which we had the most discussion about is “comfortably reaching all key elements with one hand”. Essentially speaking, this is deciding how we are designing our one-hand keyboard. The common element amongst all relevant designs is that we want to somehow move the edge keys closer to the user’s thumb. Our approaches can be generally broken down into two types: directly moving the keys on the keyboard via some form of active user motion input, and relying on some other technology, such as the gyroscope in this context, to move the keys’ positions with the help of gravity in a passive manner.

For the former type, we have two designs using the swiping motion, one being that we “squish” the entire keyboard towards a direction when the user swipes, and the other being that we “scroll” the keyboard towards one direction. The former design will display all keys at the same time regardless how far it is pushed, whereas the latter would have a portion of the keys fade out of the area of display on the keyboard, but the hidden portion would reappear if swiped back.

The gyroscope design, “the perspective keyboard”, is to utilize the perspective phenomenon that metaphorically tilts the keyboard towards a direction when the mobile device is tilted by the user. The intent is to “tilt” edge keys towards the screen, as well as the proximity of the user’s hand. The keys closer to the edge would gradually enlarge in size as well in correspondence with the perspective effect. This motion is controlled entirely by the user’s wrist so there is no need for the user to conduct any gestures on the screen. However, this design would also have a portion of the keyboard hidden from the user when the device is tilted to a certain degree, and requires the device to be at a relatively more leveled configuration for the hidden portion to reappear again.

Eventually we chose the “squish” design since the “scroll” design is deemed logically simplistic and tedious to maneuver (constant scrolling between inputs), and the gyroscope design is too radical and sophisticated, and may introduce complications during evaluation. The “squish” is a more straightforward and effective solution that moves keys to a reachable area while giving the user access to all keys at the same time. However, the tradeoff is that the keys do become narrower as the keyboard is squished, thereby raising the likelihood of text entry slips, which inevitably goes against another user requirement regarding reducing user error. Yet we accept this tradeoff because this design gives the user control over how much the keyboard is shifted in the first place, and our requirements regarding typo fixing could potentially mitigate the importance of the requirement about user error.

There was also a debate as to whether functional switch keys (upper-case switch, language switch, and numeric/punctuation keyboard switch) should be placed in a designated region, separated from other keys, near the user’s hand, or be shifted along with other keys on the one-hand keyboard. If we were to place these keys in a specific region, the tradeoff would be that although we can place them at wherever that is ergonomically comfortable for the user at our discretion, their positions may conflict with other app-specific UI elements, and may warrant some extent of overhaul of the original keyboard’s layout, which is what we want to avoid for this iteration. We can avoid this problem altogether if we just let them shift together with the other keys, at the expense of not being able to place them at more ideal locations on the keyboard interface.

When tested against the problems of text entry described in the related work section, our design addresses some of these issues. One main concern in the designs proposed in the research papers was the high learning curve to use efficiently. This is due to the designs differing from the standard QWERTY keyboard. On the other hand, our design preserves most of the QWERTY layout that many people are familiar and comfortable with. Therefore, the learning curve for our design is much lower. The only exception is the addition of new punctuation keys on the bottom right that replaces the return key. However, none of our participants mentioned using this key during the contextual inquiry, so the new keys should not have much impact on the learning curve. In addition, our design can help prevent injuries described in the related work section. The shifting feature allows users to “squish” the keyboard so that all keys are within reach of their thumbs. Users will no longer need to strain their thumbs or wrists to compensate for their lack of reach, which is one of the leading causes of injuries.

**Conclusion**

Our previous survey into one-handed text messaging on mobile devices provided foundational insights that set the stage for our subsequent research. Among the 27 participants, it was evident that one-handed typing is a prevalent method on smartphones. However, it came with its challenges: users found it less comfortable, more challenging, and slower than two-handed typing. This highlighted potential limitations in current smartphone keyboard designs and emphasized the practical importance of our research.

Building on this foundation, our detailed contextual inquiry further illuminated the differences of one-handed text messaging. We discovered that the existing keyboard layout often poses challenges, especially when users are multitasking. The challenges users face with current keyboard layouts, autocorrect features, and the act of switching between different keyboard modes have informed 6 specific user requirements. These requirements will be crucial in guiding the design of future mobile keyboards optimized for one-handed use.

Based on the specific requirements, we derived our initial design as the low-fidelity paper prototype. We also designed several design critiques and iteratively optimized our prototype. Our design mainly established and focused on the ‘sqush’ and ‘scroll’ method so users can comfortably reach all key elements with one hand. Our design also retains most of the traditional QWERTY keyboard layout, ensuring users face a minimum learning curve as well as solving main challenges in one-handed typing.

In conclusion, the knowledge generated from our initial survey, contextual inquiries, and our initial low-fidelity paper prototype serves as a robust foundation for future work. It not only provides insights and potential solutions into the current challenges users face but also guiding design and evaluation processes. We are optimistic that future researchers and designers will leverage these findings, leading to innovations that cater to the evolving needs of one-handed mobile device users.