FIT3036 Computer Science Project
Design of keyboards and other interfaces for users with reduced capacity

Final Report
TNK – The Numeric Keyboard

Student Name	: Reynald Nixon
Student Contact	: rnix4@student.monash.edu
Student ID	: 27022722
Supervisor Name	: Rasika Amarasiri
Word Count	: 3483 Words

Abstract: Keyboard design often assumes that user will type in with two hands. However, there are users with reduced capacity such as users using one hand to type with the keyboard. These users will not be able to type in the normal QWERTY keyboard layout efficiently the layout was designed for people with 2 hands. This paper builds upon the research of existing options that are available for users to type using one hand. It starts with a basic introduction of what the project is about and the background of the project. Based on the research, a new keyboard layout called The Numeric Keyboard was developed in order to cater to users who uses one hand to type. The new layout has some features that will be explained further in the paper to be considered as efficient. The Numeric Keyboard's quality will also be explained along with its future works that will make The Numeric Keyboard a new alternative for one hand typing layout.

Keywords: Keyboard, Usability, Numeric Keypad, Keyboard Layout

Table of Contents

1.0. Introduction

1.1. Project Objectives

Keyboard commands such as Ctrl + Alt + Del and other commands often assume that the user is working with two hands (Dowe, 2018). Users with reduced capacity such as someone who injured one of their hand are only able to work with one hand and will find it hard to do the keyboard commands that needs two hands. The objective of this project is to design a new keyboard layout or update an existing keyboard layout to be able to adapt to users with reduced capacity so they are able to interact more efficiently with the new keyboard interface. As the project scope was too broad, the scope of the project was narrowed down to creating or updating a keyboard layout for people with one hand to interact efficiently with the interface.

1.2. Project Requirements

1.2.1. Functional Requirements

The project requires the developer to create a software to aid users with reduced capacity to type efficiently. It is also focused on the software side rather than the hardware side therefore we do not need to create a new hardware. The project can be done in any programming language and the developer is able to use any libraries to support the project. Research is also needed to find out what existing options are currently available to tackle the problem so that our project will not overlap with existing projects.

1.2.2. Non-Functional Requirements

1.2.2.1. Documentation

Included in the project to ensure that both users and developers understand the project. Also ensures readability.

1.2.2.2. Efficient and Effective

Users will use one hand and the travel distance between each letter will be less compared to QWERTY keyboard layout

1.2.2.3. Open Source Project

Ensure code is modifiable by other developer to ensure code is up to date

1.2.2.4. Scalable

Project can be extended to serve different purposes or could be updated for more features.

1.2.2.5. Robustness

Ensure the project can withstand heavy testing and will always be in a good quality after every usage.

1.2.2.6. Usability

The project will cater to user with either left-hand or right-hand injury. Also,it can be used by people with two hands.

1.3. Project Constraints

First constraint identified in doing this project will be the time constraint given to finish the project. The project will be done in the time of 12 weeks: first 6 weeks for research and writing a proposal for the project that will be done, the next 4 weeks is to finish the project which includes the design, test plan, implementation and testing of the project itself. The last 2 weeks is for presentation and doing the test and final report for the project. Second constraint will be the labour constraint because the project must be done individually. The topic that was given which is designing a keyboard is broad and can be interpreted in many ways. Having 2 or more members doing a project will help the brainstorming process for ideas and enables the final product to be delivered in its full potential. Combining the two constraints above, there is a potential that the final product delivered will not be in its full potential as proposed in the project proposal as the project might encounter some obstacles in its development phase.

2.0. Background

2.1. Existing Academic Literature

Because the project scope has been narrowed down to users with one-hand injury, the first thing to do was to search for existing alternatives to QWERTY keyboard for one-hand typing. The option that was common is using a one-hand Dvorak layout or the half-QWERTY keyboard layout. Upon further research, an academic literature of a project for a numeric keypad based text input called TNT was discovered. TNT aims to be a text input for television remote (Ingmarsson, Dinka & Zhai, 2004). The discovery of TNT leads to an idea of creating a new keyboard layout using only numeric keypad which is reachable with one hand.

2.2. Datasets Used in the Project

The datasets used in this project is the statistics of most used letters in English words according to the research by oxford dictionary ("Which letters in the alphabet are use... | Oxford Dictionaries", 2018). The reason this dataset is needed in this project is because the most used letter in English words will map into the least number of numeric keys that the users need to press to produce the letter.

2.3. Risk Analysis

2.3.1. Risk Register

ID	Risk Description	Likelihood	Impact	Composite Index	Mitigation	Contingency
1	Laptop doesn't have numpad	5	3	15	Have an optional mapping in the main keyboard that imitates numeric keypad	-
2	User not remembering key combination	9	5	45	Providing references in the software	-
3	Library not supported	4	5	20	Use library from an active developer	Try to find similar library and implement it

4	JavaScript function is removed in an update	2	8	16	Do not use an uncommon function that aren’t frequently used by developers	Try to follow with the update and see if the functions can be updated manually
5	Hardware used to develop the project is broken	2	8	16	n/a	Use campus lab computer or borrow a device from colleagues
6	Project data is corrupted	1	10	10	Regularly save and back up work files on a cloud storage	Try to recover data using some sort of software
7	Developer got sick	2.5	10	25	Ensure developer got enough rest	Drink medicine to recover
8	Lack of experience in software development	3	6	18	Do research on resources before building the project	Get help from colleagues or supervisor
9	Developer other commitments	3	8	24	Have a good time management plan for doing each commitment	Do what is due first and ensure that commitment are done on time

2.3.2. Actual Risk Encounters

2.3.2.1. Hardware used for development broke down in the middle of development phase

2.3.2.2. Developer got sick

Because of the 2 risks encountered in the development phase, the software development was delayed a few times and some extra functionalities that were supposed to be implemented was not implemented such as word suggestion and text prediction.

2.4. Resource Requirements

2.4.1. Hardware Requirements

2.4.1.1. PC or laptop with a numeric keypad if possible

2.4.1.2. Keyboard

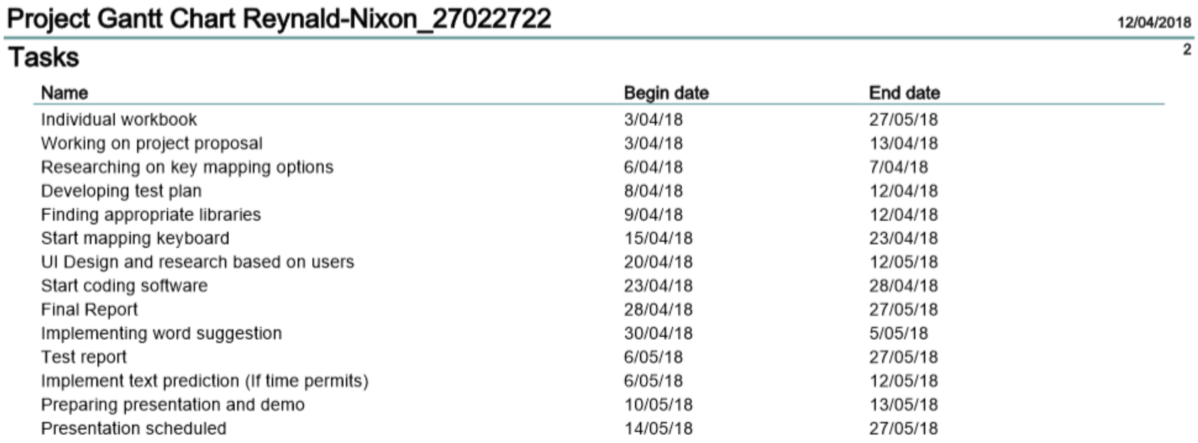
2.4.1.3. Internet connection

2.4.2. Software Requirements

2.4.2.1. JavaScript programming language
2.4.2.2. IDE for JavaScript such as Visual Studio Code
2.4.2.3. Node.js
2.4.2.4. Web browser such as Google Chrome
2.4.2.5. Various libraries such as Socket.io, Express, etc

2.5. Project Timeline and Tasks

2.5.1. Initial Timeline and Tasks

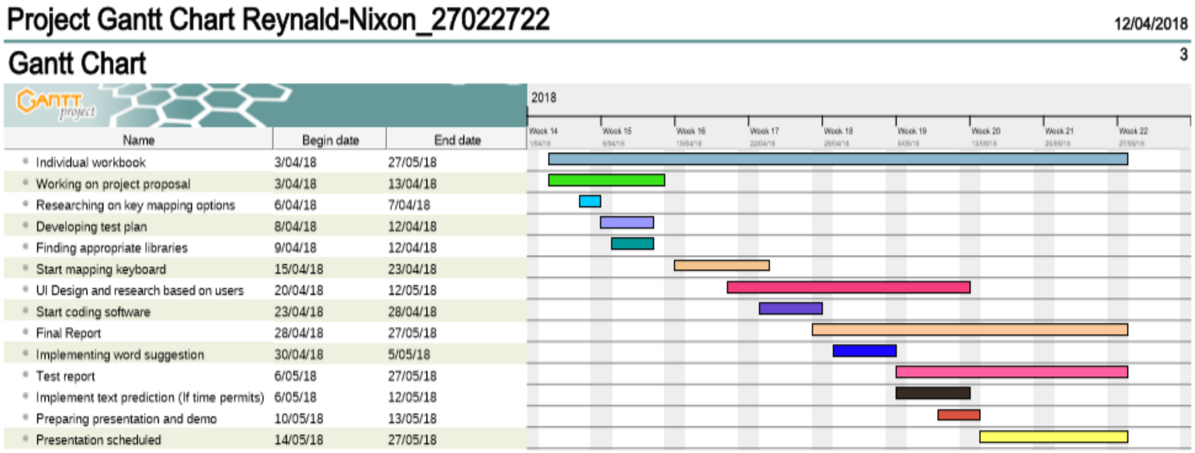


Project Gantt Chart Reynald-Nixon_27022722 12/04/2018

Tasks 2

Name	Begin date	End date
Individual workbook	3/04/18	27/05/18
Working on project proposal	3/04/18	13/04/18
Researching on key mapping options	6/04/18	7/04/18
Developing test plan	8/04/18	12/04/18
Finding appropriate libraries	9/04/18	12/04/18
Start mapping keyboard	15/04/18	23/04/18
UI Design and research based on users	20/04/18	12/05/18
Start coding software	23/04/18	28/04/18
Final Report	28/04/18	27/05/18
Implementing word suggestion	30/04/18	5/05/18
Test report	6/05/18	27/05/18
Implement text prediction (If time permits)	6/05/18	12/05/18
Preparing presentation and demo	10/05/18	13/05/18
Presentation scheduled	14/05/18	27/05/18

Figure 1. Initial tasks to be done



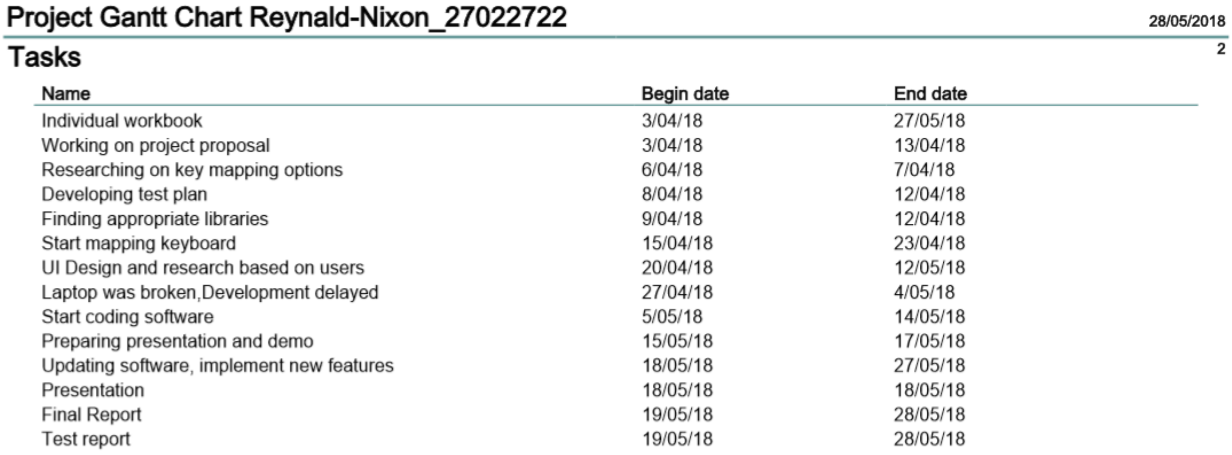
Project Gantt Chart Reynald-Nixon_27022722 12/04/2018

Gantt Chart 3

Name	Begin date	End date
Individual workbook	3/04/18	27/05/18
Working on project proposal	3/04/18	13/04/18
Researching on key mapping options	6/04/18	7/04/18
Developing test plan	8/04/18	12/04/18
Finding appropriate libraries	9/04/18	12/04/18
Start mapping keyboard	15/04/18	23/04/18
UI Design and research based on users	20/04/18	12/05/18
Start coding software	23/04/18	28/04/18
Final Report	28/04/18	27/05/18
Implementing word suggestion	30/04/18	5/05/18
Test report	6/05/18	27/05/18
Implement text prediction (If time permits)	6/05/18	12/05/18
Preparing presentation and demo	10/05/18	13/05/18
Presentation scheduled	14/05/18	27/05/18

Figure 2. Initial timeline

2.5.2. Timeline and Tasks After Software Development



Project Gantt Chart Reynald-Nixon_27022722

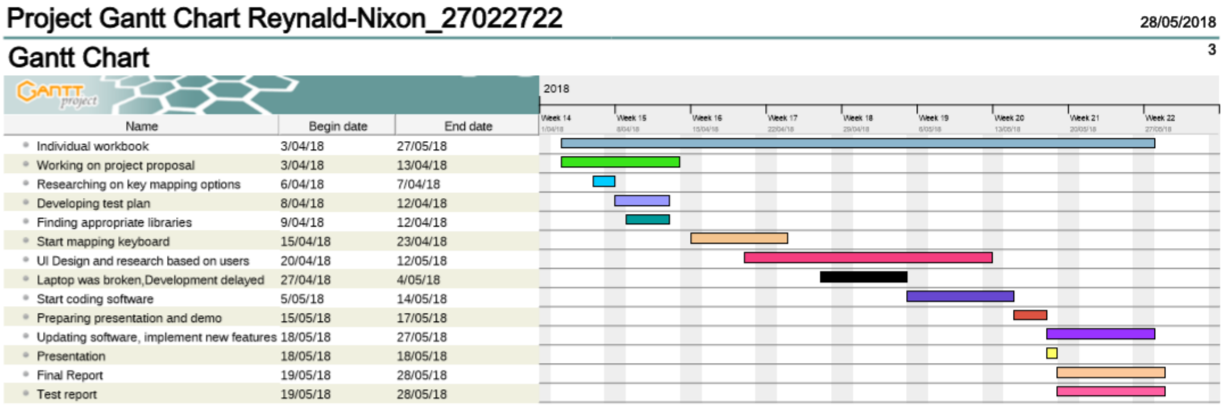
28/05/2018

Tasks

2

Name	Begin date	End date
Individual workbook	3/04/18	27/05/18
Working on project proposal	3/04/18	13/04/18
Researching on key mapping options	6/04/18	7/04/18
Developing test plan	8/04/18	12/04/18
Finding appropriate libraries	9/04/18	12/04/18
Start mapping keyboard	15/04/18	23/04/18
UI Design and research based on users	20/04/18	12/05/18
Laptop was broken,Development delayed	27/04/18	4/05/18
Start coding software	5/05/18	14/05/18
Preparing presentation and demo	15/05/18	17/05/18
Updating software, implement new features	18/05/18	27/05/18
Presentation	18/05/18	18/05/18
Final Report	19/05/18	28/05/18
Test report	19/05/18	28/05/18

Figure 3. Tasks to be done after software development



Project Gantt Chart Reynald-Nixon_27022722

28/05/2018

Gantt Chart

3

Name	Begin date	End date
Individual workbook	3/04/18	27/05/18
Working on project proposal	3/04/18	13/04/18
Researching on key mapping options	6/04/18	7/04/18
Developing test plan	8/04/18	12/04/18
Finding appropriate libraries	9/04/18	12/04/18
Start mapping keyboard	15/04/18	23/04/18
UI Design and research based on users	20/04/18	12/05/18
Laptop was broken,Development delayed	27/04/18	4/05/18
Start coding software	5/05/18	14/05/18
Preparing presentation and demo	15/05/18	17/05/18
Updating software, implement new features	18/05/18	27/05/18
Presentation	18/05/18	18/05/18
Final Report	19/05/18	28/05/18
Test report	19/05/18	28/05/18

Figure 4. Timeline after Software Development

The tasks to be done and the timeline has changed from the initial schedule (Figures 1 and 2) into the schedule after software development (Figures 3 and 4). Some tasks for adding functionalities to the project such as word suggestion and text prediction has been abandoned because of the encounter of some actual risk such as developer becoming sick and laptop broken for a week. This causes some tasks to be delayed especially when the laptop is broken, the developer's data is in the laptop and it is being repaired for a week. An additional task has been added into the new schedule (updating software, implement new features) because after the presentation, the developer notices a few bugs and updated the software, also adding new features to the software such as backspaces and enter key.

3.0. Method

3.1. Methodology

The methodology used for developing this project is Agile methodology. In agile methodology, changes in uncertain environment can be tackled through collaboration between self-organizing, cross-functional teams utilizing the appropriate practice ("What is Agile Software Development?", n.d.). What is done was the supervisor will discuss the topics in class because most likely the developer and colleagues will take the same projects. Meetings between peers and supervisor are done weekly so everyone will get an idea of what to build and to ensure that no project are the same. In the meetings as well, the supervisor asks for people who have done some work to give a presentation in order to get feedback from colleagues. To maintain the schedule, in Agile methodology, a checklist was made with each task divided into small sub tasks. After each subtask are done, it can be checked and last, the main task will be checked slowly. The main point of this methodology is to complete a small task in each iteration and in the end, the big tasks will be accomplished.

3.2. Internal Design

The main part of the completed project will be the mapper function in the server side and the use of Socket.io to transfer data between client and server in real time. The activity diagram below (Figure 5) describes how the system will work together. The user will input some key combinations that will be send to the server side whenever the client presses enter key and the combination will be checked in the server side by the mapper function. If the combination is valid then socket.io will send the corresponding letter into the client side. However, if the key combination is incorrect, Socket.io will send a reset respond to the client and client's current input will reset. If the user has finished typing the words he/she wants, the user can close the text editor, if not finished, then user can continue typing in key combination. The transferral of data between client and server will always go through Socket.io and the checking of key combination will always be done in server side by the mapper function.

Activity Diagram

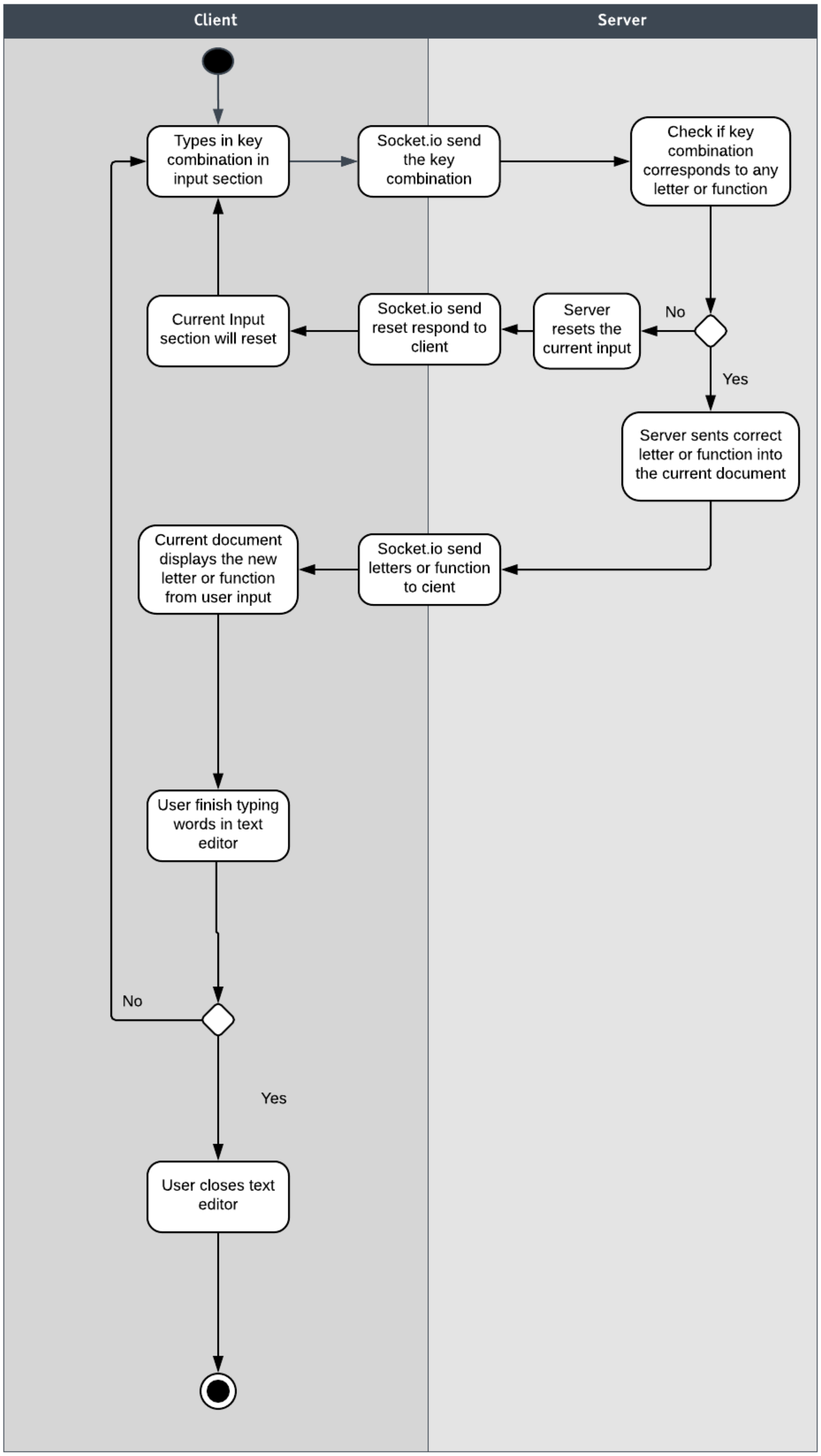

Client
Server
Types in key combination in input section
Socket.io send the key combination
Check if key combination corresponds to any letter or function
No
Server resets the current input
Socket.io send reset respond to client
Current Input section will reset
Yes
Server sents correct letter or function into the current document
Socket.io send letters or function to cient
Current document displays the new letter or function from user input
User finish typing words in text editor
No
Yes
User closes text editor

Figure 5. Activity Diagram for The Numeric Keyboard

3.3. Software Architecture

The high-level class structure of the project is described below in a UML class diagram. Explanation of each class is explained as the following:

3.3.1. **Keyboard**: Has an attribute of numerical keys to be pressed by user and can send 1 or many keys to the translator system using the send keypress function. It also has the function of enter to indicate the server to check the key combination

3.3.2. **Translator System**: Contains the dictionary of number, letters, symbols and function combination for translating numerical values into letters (mapper function) and is located in the server side. The mapper function will check if a combination is valid or not and will produce a result for socket.io to send to the document.

3.3.3. **Socket.io**: is the main mode of transport between the server and client. It sends data between server and client in real-time.

3.3.4. **Text Editor**: Where the letters will be displayed and is located in the client side, where user will interact with the document. It has an attribute of document ID and title, also has the current letters/words typed. Its function is mostly to construct sentences, close document, receiving result from translator via socket.io

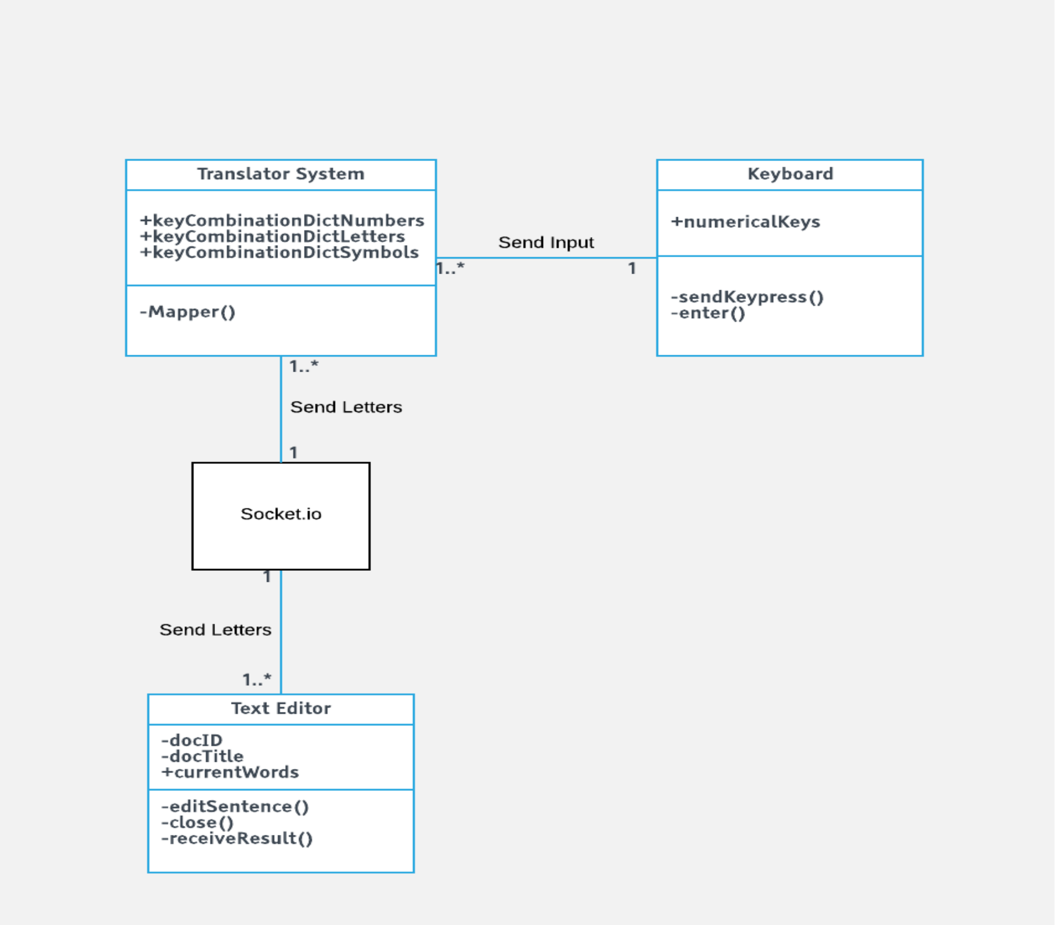


Figure 6. UML Class Diagram of The Numeric Keyboard

3.4. Key Algorithms

3.4.1. Mapper function algorithm pseudocode

1. Mapper receive a letter input from client via socket.io
2. Mapper push letter into array, pass the letter back to socket.io to display in current input
3. Mapper receives enter command from user via socket.io
4. Mapper joins the array and puts string result in a variable, mapper resets array for next combination
5. Mapper checks string result with key combination dictionary
    - If combination is found, convert into letter and ask socket.io to send result to client to display in current document
    - Else, combination not found, clear variable containing the string result, ask socket.io to send reset command to client, client-side current input will be reset
6. Repeat process until user finish typing in text editor

3.4.2. Socket.io algorithm pseudocode

1. Socket.io send data based on events and is placed in both client and server
2. It works as a transport system between client and server
3. You can think of socket.io as a station for a train that only goes between 2 stations
4. On a certain event that is triggered, socket.io from client will send an event that will be received by the server. After doing the mapper function, the server will send an event that will be received by the client side.

3.4.3. The Numeric Keyboard pseudocode

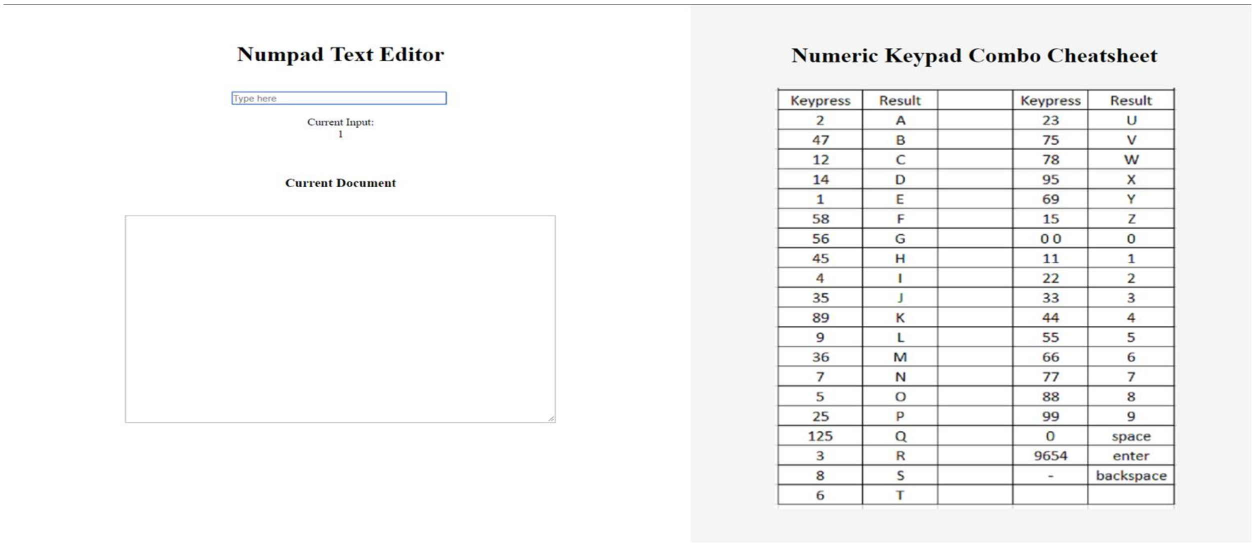
1. User input a key in the client side, socket.io sends that key into server that keeps the key in an array. Server prompts socket.io to send the key back to the client to be displayed in current input
2. User continues sending key combination until enter key is pressed, socket.io send enter command to the server, telling server to start mapping function.
3. After mapping function as described in section 3.4.1, mapper will send result to the client via socket.io
4. Client side will either get a reset request for current input from socket.io, or client side will receive a letter in current document.

4.0. Results

4.1. TNK – The Numeric Keyboard

The result of this project is the creation of The Numeric Keyboard, which is a new keyboard layout. TNK will be a custom mapping of a normal QWERTY keyboard keys into the numerical keypad, which will increase the usability of one-hand control as the numeric keypad is in the range of one hand. The reason why numeric keypad was chosen is because when typist is using the numeric keypad, they tend to use only 1 hand which is usually the right hand. Combination of numerical keys entered (called combo) will result to letter. The mapping itself for the combo is based on the statistics of letters in English words. The common that the letter appears in the English words, the less keys that user will press to produce that letter. The mapping is also based on the statistics where in QWERTY keyboard layout, typist will often hurdle through the middle row of the keyboard (home row) (Diamond, 1997). Therefore, the mapping reduces the need to hurdle over the middle row of the numeric keypad because the mapping is designed to end up in the middle row for most of the time.

4.2. Features of TNK – The Numeric Keyboard



Numpad Text Editor

Current Input:
1

Current Document

Numeric Keypad Combo Cheatsheet

Keypress	Result		Keypress	Result
2	A		23	U
47	B		75	V
12	C		78	W
14	D		95	X
1	E		69	Y
58	F		15	Z
56	G		0 0	0
45	H		11	1
4	I		22	2
35	J		33	3
89	K		44	4
9	L		55	5
36	M		66	6
7	N		77	7
5	O		88	8
25	P		99	9
125	Q		0	space
3	R		9654	enter
8	S		-	backspace
6	T			

Figure 7. The User Interface for TNK – The Numeric Keyboard

As explained earlier in the report, the custom mapping will be based from the oxford dictionary statistics of most used letters in English words. Reduction in keys being pressed (maximum of 4 key combination to be pressed) and the mapping that is customized to end up in the middle row most of the time means reducing the strain in fingers travelling to type letters. The TNK also have an alternative mapping for users that does not have numeric keypad in their hardware, which is a mapping in the middle of the QWERTY keyboard that imitates the numeric keypad shown in Figure 8 below.

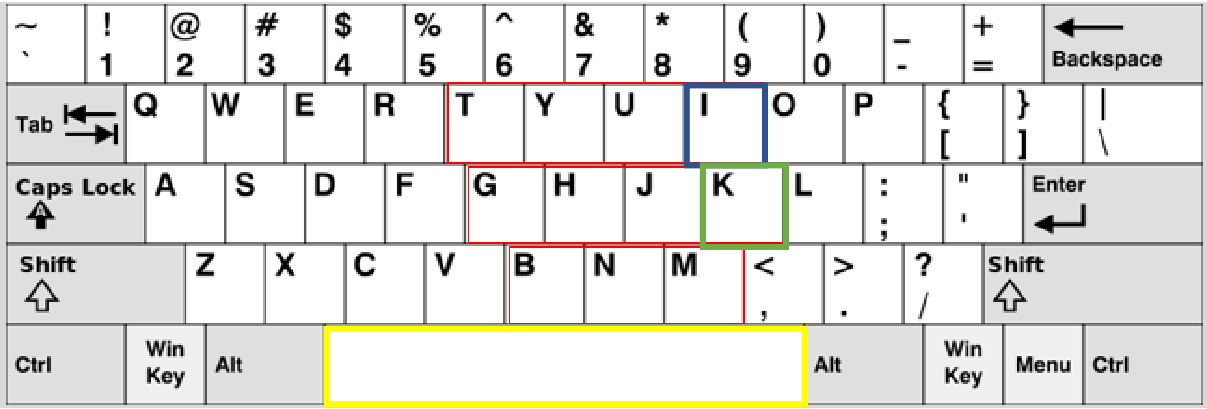

~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Win Key Alt Alt Win Key Menu Ctrl

Figure 8. Alternative mapping for the TNK layout

The alternative mapping uses 12 keys that imitates the numeric keypad's position. The red keys (B,N,M,G,H,J,T,Y,U) imitates keypad 1-9 respectively, the blue key (I) imitates the minus key in the keypad, the green key (K) imitates enter in the keypad and last, the yellow key (Spacebar) imitates 0 in the keypad.

4.3. Performance of TNK – The Numeric Keyboard

In terms of time complexity, the mapper function will be O(m) where m is the number of keys mapped to the numerical keypad. Here, because number of keys mapped is 39 keys, therefore the time complexity will become O(39) for worst case and O(1) for best case which is considered O(1) or constant time complexity. For space complexity, it will be O(n) where n is the number of number kept in the array on server side until the array is reset to empty.

5.0. Analysis & Discussion

5.1. Tests Methods for TNK

Unit testing and system testing are done to evaluate the performance of TNK. For unit testing, it is testing the main modules for the TNK which are the mapper function in the server side and the socket.io that works as a transport system to send data between client and server in real-time. For system testing, there are number of tests that were done to evaluate TNK's performance which are acceptance testing, performance testing, usability testing, stress testing, beta testing, integration testing and function testing. The test was done by asking 5 of my colleagues to be the beta testers for the project. Detailed explanation for each testing is available in a separate test report attached with this final report.

5.2. Analysis & Discussion of TNK's Quality

In terms of typing speed, the expected words per minutes (WPM) of people typing with TNK layout is 25 WPM which is 15 WPM less than an average typist with a QWERTY keyboard layout (Karat, Halverson, Horn & Karat, 1999). In the performance testing section of the test report, the 5 beta testers and myself manage to achieve an average of 20 WPM which is below the expected result. After discussion with each of the beta testers, we have reach a conclusion that the typing speed is below the expected result is mainly because of the key combination that is not familiar among the users. When using the app, they need to refer to the cheat sheet most of the time and that takes a bit of time. The beta testers all agree that if they have memorized the key combination, they can reach the expected result of 25 WPM or maybe more.

In terms of efficiency, the beta testers have reached a unanimous decision of agreeing that the keyboard layout is efficient to be used with one-hand. The reason is because the numeric keypad is reachable by one-hand and they only need small finger movements to type between each numeric keypad. The fact that TNK also have a feature that the combination will most likely end in the middle row also adds the efficiency factor as key presses rarely hurdles over the middle row. TNK also met the business requirement of creating a keyboard layout for users with either left or right hand to type with one hand.

In terms of usability, beta testers also agreed that TNK project has a good usability factor. Different users can use TNK because of the feature where if a user only uses one hand, he/she can use the input button with the numeric keypad or the alternative input method, and if the user wants to type in the QWERTY layout or a user wants to type with two hands, he/she can still use the TNK by directly typing in the current document using normal keyboard layout. Overall, the beta testers concluded that TNK has the potential to be used as an alternative if the user must type using only one hand.

6.0. Future Work

6.1. Implement a more efficient mapping

Even though the mapping that is available now is efficient according to the beta testers, the developer believes that the mapping can be improved to ensure less distance travel between keys and by further research of taking the statistics of English words, combinations can be made to always end up in the middle row after words are typed.

6.2. Make the keyboard layout available for use in the device

Currently, the keyboard layout is only available to be used in the text editor provided by the app itself. After user is finish using the text editor, they need to copy the whole document and paste it into their desired text editor such as notepad or Microsoft word. In the future, it will be easier if TNK is available to be used anywhere in the PC or laptop and works like the QWERTY keyboard

6.3. Map more keys and functionalities into the keypad

In this project, 39 keys combination consisting of 26 capital letters, 10 numbers, space, enter and backspace managed to be mapped into TNK. In the future, more keys and functionalities will be added to the TNK such as lowercase letters, caps lock modifier and if possible, function that requires two hands such as Ctrl + Alt + Del could be mapped into some key combinations as well.

6.4. Implementation of word suggestion and text prediction

Currently, to create a word or sentences, users need to type in each letter one by one. If word suggestion future is implemented, it will save some time for users to type in words because after typing a few letters, the system will generate some suggestion and users can pick the word that they want. Also by introducing text prediction, users will have an autocorrect function that helps correct mistakes of misspelled words.

6.5. Improvement of UI design

TNK now has a simple UI design as seen in Figure 7 which consists of a screen split into two. The left side has a user input area with the current input information below it. The document is also on the left side while on the right side, the cheat sheet is displayed to help user enter key combinations. The UI design can be improved to attract more users by adding more elements and making it colourful. It will also help if users are notified when they enter wrong key combination which is not available now.

6.6. Option for user to create own mapping

Currently, the mapping is already decided and users are forced to use the available mapping. It will be better if there exist an option for users to create their own custom mapping that suits their own style.

6.7. Potential to be used in another platform besides computer

Because TNK was based on TNT, there’s a potential that TNK might be used for an input method for a television.

7.0. Conclusion

The project given manage to result in a new keyboard layout called the TNK – The Numeric Keyboard which is based on a previous research called TNT. The Numeric Keyboard helps cater to people who injured one of their hands or people who only have one hand. The main feature for the TNK lies on the efficient mapping of the QWERTY keyboard into a small area in the numeric keypad. It also reduces the strain from typing because user's will not need to hurdle over the middle row often. Last feature of the TNK is where the keyboard mapping is based on the letter statistics in English words in which most used letter will map to the least amount of keys needs to be pressed. TNK was also tested by some beta testers, who were satisfied with the result of TNK layout. In conclusion, with future works in line for the project, TNK has the potential to be the layout that people will use for typing with one hand.

8.0. Bibliography

[1] Dowe, D. (2018). FIT3036_DDowe_1stSemester2018ProjectList.pdf. Retrieved 28 May 2018, from Monash University FIT3036 Computer Science Project Moodle Page. Monash University.

[2] Ingmarsson, M., Dinka, D., & Zhai, S. (2004). TNT: a numeric keypad based text input method. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 639-646). Vienna, Austria. Retrieved from https://doi-org.ezproxy.lib.monash.edu.au/10.1145/985692.985773

[3] Which letters in the alphabet are used most often? | Oxford Dictionaries. (2018). Retrieved 25 March 2018, from https://en.oxforddictionaries.com/explore/which-letters-are-used-most

[4] What is Agile Software Development?. Retrieved from https://www.agilealliance.org/agile101/

[5] Diamond, J. (1997). The Curse of QWERTY | DiscoverMagazine.com. Retrieved from http://discovermagazine.com/1997/apr/thecurseofqwerty1099/

[6] Karat, C., Halverson, C., Horn, D., & Karat, J. (1999). Patterns of entry and correction in large vocabulary continuous speech recognition systems. In Proceedings of the SIGCHI conference on Human Factors in Computing Systems (pp. 568-575). Pittsburgh, Pennsylvania, USA. Retrieved from https://doiorg.ezproxy.lib.monash.edu.au/10.1145/302979.303160

[7] Nixon, R. (2018). Project Proposal "Design of keyboards and other interfaces for users with reduced capacity". Monash University.

[8] Nixon, R. (2018). Test Report "Design of keyboards and other interfaces for users with reduced capacity". Monash University.

[9] https://socket.io/

[10] FIT3140 2nd semester of 2017 Lecture slides and Labs. Monash University

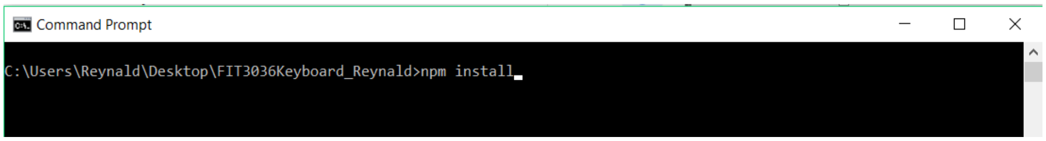
9.0. Appendices

9.1. Production and Deployment Instructions

9.1.1. Instructions to run the software

1. Users will need to have Node.js installed in their device which can be obtained from the following website: https://nodejs.org/en/

2. After installing Node.js, next step will be installing all the dependencies, which can be done by opening a command prompt and going into the directory of the project, after that type the command "npm install" to install all dependencies. The picture shows the scenario where the software folder is in C:\Users\Reynald\Desktop\FIT3036Keyboard_Reynald

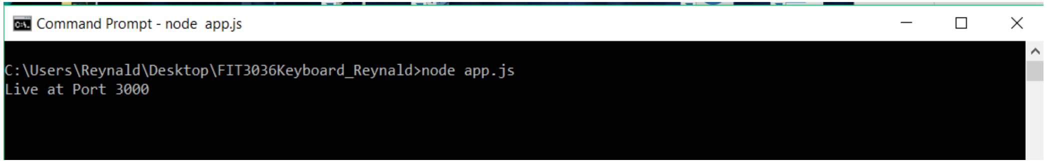


```
C:\Users\Reynald\Desktop\FIT3036Keyboard_Reynald>npm install
```

After running the command, all dependencies will be installed because the package.json file in the folder contains every dependencies that the user needs in order to run the software.
The following dependencies are needed for the software to run:

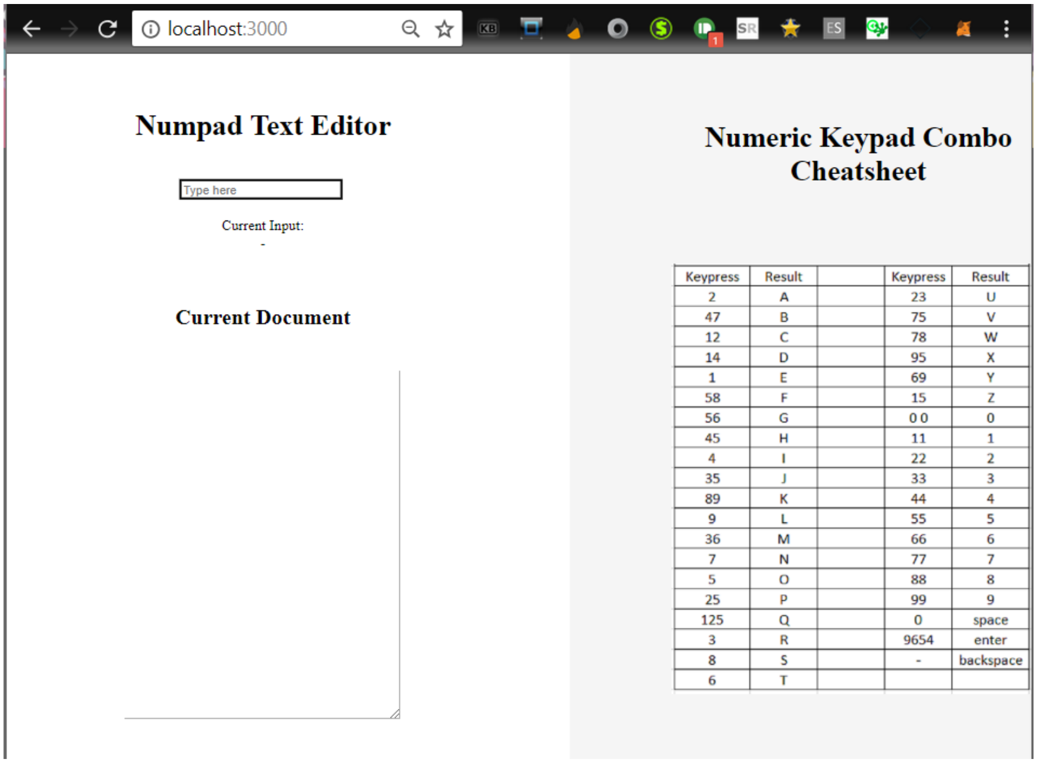
- Editor.html that functions as the text editor page
- 404.html functions as a page that user will be redirected if they type in a wrong URL
- App.js functions as the server
- Package.json that contains the software dependencies
- Public folder that contains the editor.css for client-side html style and cheat sheet picture

3. After running "npm install" command, you are now ready to run the software. You can run the software by typing "node app.js" as shown below. After entering the command, the console will reply with the message "Live at port 3000"



```
C:\Users\Reynald\Desktop\FIT3036Keyboard_Reynald>node app.js
Live at Port 3000
```

4. After seeing the message above, you can open any browser and type in the following address “localhost:3000”. The software is now ready to be used.



Numpad Text Editor

Type here

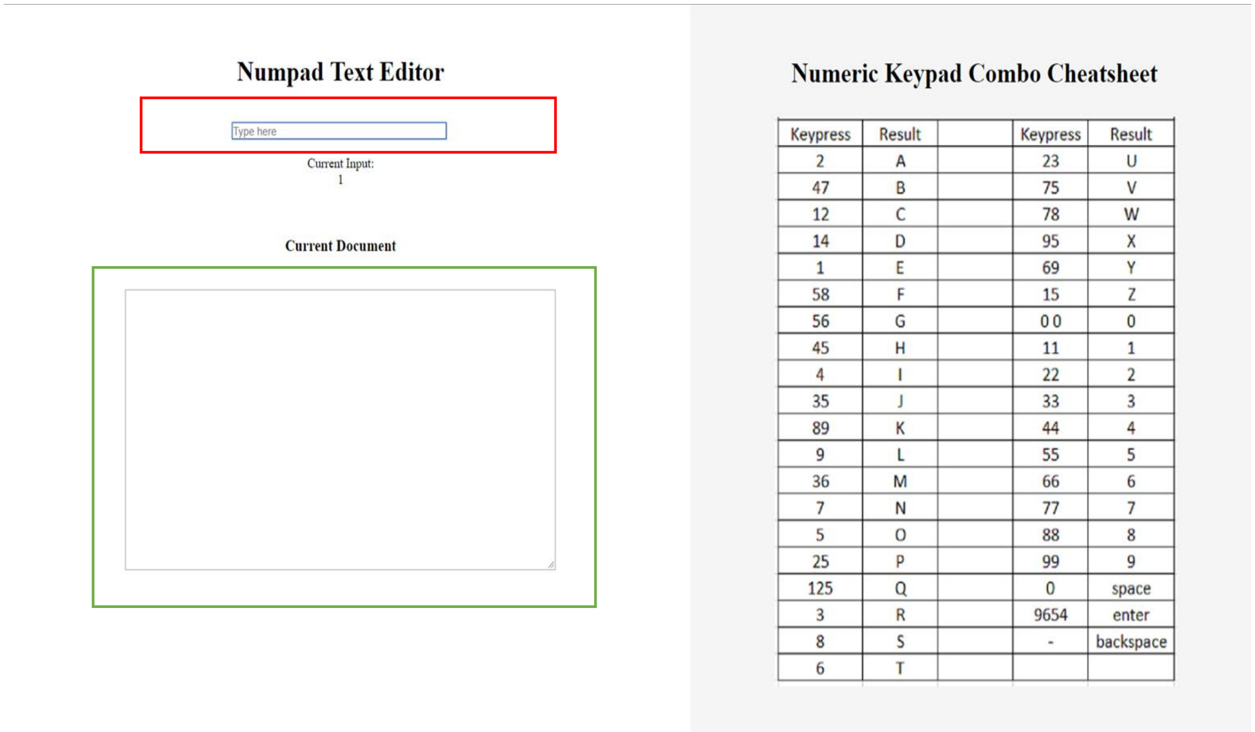
Current Input:
-

Current Document

Numeric Keypad Combo Cheatsheet

Keypress	Result		Keypress	Result
2	A		23	U
47	B		75	V
12	C		78	W
14	D		95	X
1	E		69	Y
58	F		15	Z
56	G		0 0	0
45	H		11	1
4	I		22	2
35	J		33	3
89	K		44	4
9	L		55	5
36	M		66	6
7	N		77	7
5	O		88	8
25	P		99	9
125	Q		0	space
3	R		9654	enter
8	S		-	backspace
6	T			

9.2. User Interface



Numpad Text Editor

Current Input:
1

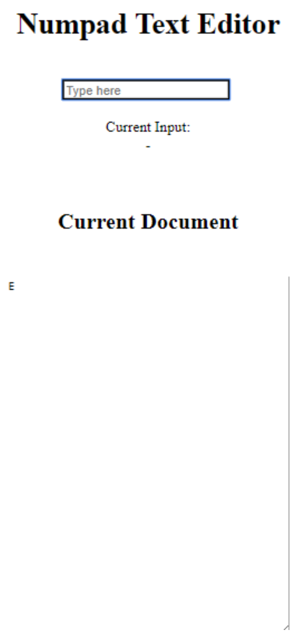
Current Document

Numeric Keypad Combo Cheatsheet

Keypress	Result		Keypress	Result
2	A		23	U
47	B		75	V
12	C		78	W
14	D		95	X
1	E		69	Y
58	F		15	Z
56	G		0 0	0
45	H		11	1
4	I		22	2
35	J		33	3
89	K		44	4
9	L		55	5
36	M		66	6
7	N		77	7
5	O		88	8
25	P		99	9
125	Q		0	space
3	R		9654	enter
8	S		-	backspace
6	T			

Above is the picture of the user interface that the user will see after running the app. There are two input fields where user can interact with which is marked above by the red box and the green box. Instructions to use the app will be provided below:

- Red square: This is where users can use one hand to input the key combinations. Users can press once on the box and start typing in the key combination by referring to the cheat sheet in the right side. Each time user presses a key, the key that the user pressed will be displayed in the current input section. To convert the current input into a letter, user needs to confirm their key combination by pressing the “enter” key beside numeric key “3” or “k” if using the alternative mapping. After confirming, the current input section will display “-“ and the current document section will have the corresponding letter (“E”) appear as displayed below:



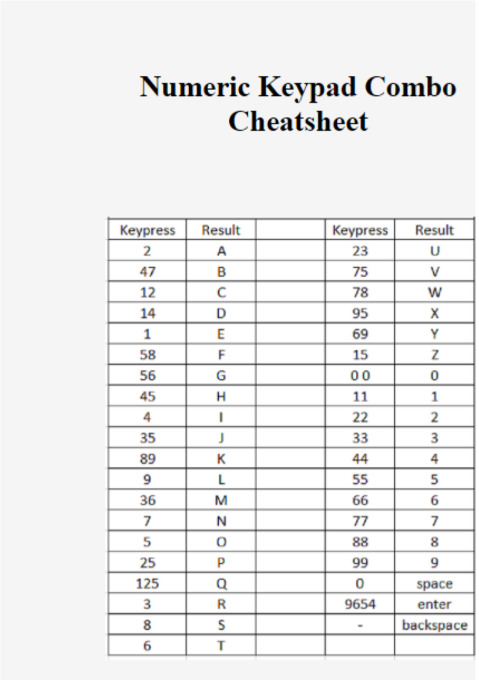
Numpad Text Editor

Current Input:

-

Current Document

E



Numeric Keypad Combo Cheatsheet

Keypress	Result		Keypress	Result
2	A		23	U
47	B		75	V
12	C		78	W
14	D		95	X
1	E		69	Y
58	F		15	Z
56	G		0 0	0
45	H		11	1
4	I		22	2
35	J		33	3
89	K		44	4
9	L		55	5
36	M		66	6
7	N		77	7
5	O		88	8
25	P		99	9
125	Q		0	space
3	R		9654	enter
8	S		-	backspace
6	T			

Extra information about red input section:

- if the user types in the wrong key combination and wishes to reset the current input, just type in a key combination that is not available in the cheat sheet and press enter, that way the current input will be reset as the key combination doesn't exist.
- Each time user wants to convert a letter or do a function, user needs to confirm by pressing enter key. For example if user has typed in "HELLOOO" and wishes to delete the 2 "O"s, user needs to press the "-" key then enter twice in order to delete the 2 "O"s. Same thing goes with spaces, if a user wants to have 3 spaces after a word, the user needs to press "0" then enter key 3 times to produce 3 spaces.
- Other keys besides the keys mapped won't work in the red input section.

- Green square: is the current document section of the text editor. User can interact with normal QWERTY keyboard and can use the keyboard normally in this square. It is used to cater people who wishes to use the app normally so that different users will be catered when using the software.

Example scenario on software usage:

For a user to type in the word "HELLO", the user needs to do the following:

1. Press 4 and 5 then enter key
2. Press 1 then enter key
3. Press 9 then enter key
4. Press 9 then enter key
5. Press 5 then enter key
6. The user will see HELLO typed in the current document

9.3. Risk Register Impact and Probability Scale

Impact Table

Score	**Description**
1-2	Limited
3-4	Low
5-6	Moderate
7-8	High
9-10	Extreme

Probability Table

Score	**Description**
1-2	Not likely
3-4	Low
5-6	Moderate
7-8	High
9-10	Expected

*Composite Index score is out of 100. the higher the score, the higher priority of attention needed