Rozdział 1

Analiza projektowa

Rozdział poświęcony jest analizie wybranych technologii oraz wymagań projektu. Przedstawiono również wstępną analizę dotyczącą przypadków użycia oraz metod testowania produktu.

1.1 Wybrane technologie

Krótki opis teoretyczny wybranych technologii - C++ oraz bibliotek QT.

1.1.1 C++

C++ jest powszechnie stosowanym językiem programowania, będącym potomkiem języka C. Wprowadzono w nim szereg udogodnień.C++, w porównaniu z C, zapewnia dokładniejsze sprawdzanie typów danych, wspiera abstrakcje, programowanie obiektowe (z tego względu mówi się o nim jako o języku pseudo-obiektowym), programowanie uogólnione i proponuje więcej styli programistycznych. Do wspieranych założeń programowania obiektowego należą polimorfizm, enkapsulacja i dziedziczenie. Nowsza wersja C - C++, posiada również bardzo dużą ilość bibliotek, których wykorzystanie znacznie ułatwia jego impletmentację. W skład klasy tego języka wchodzi plik wykonywalny (*.cpp) oraz nagłówkowy (*.h). W plikach nagłówkowych klas definiuje się wszystkie metody oraz zmienne. Głównymi powodami do tworzenia oddzielnych plików nagłówkowych są **?**:

- przyspieszenie czasu kompilacji programu. Pliki nagłówkowe nie wymagają re-kompilacji, o ile nie nastąpiła w nich zmiana, co skraca czas kompilacji w czasie rozwiania dużych projektów,
- organizacja struktury kodu,
- rozdzielenie interfejsów od implementacji.

Przykładowy plik nagłówkowy klasy ButtonOperator:

```
#ifndef BUTTONOPERATOR_H
#define BUTTONOPERATOR_H
using namespace std;
#include <QtWidgets>
#include <QStringList>



class ButtonOperator : public QPushButton
{
    Q_OBJECT

public:
   explicit ButtonOperator(QWidget *parent=0);
       void setDisplayList(QStringList sDisplayList);
    QStringList getDisplayList();
    void setSpecial(bool sIsSpecial);
    bool getSpecial();
    bool getHover();


protected:
    void hoverEnter(QHoverEvent *event);
    void hoverLeave(QHoverEvent *event);
    void hoverMove(QHoverEvent *event);
    bool event(QEvent *event);

private:
    bool isSpecial;
    bool isHovered;

    QStringList displayList;

};


#endif // BUTTONOPERATOR_H
```

Widać w nim jasno wydzielone sekcje modyfikatorów, typy zmiennych i metod oraz ich argumenty.

1.1.2 QT

QT jest zespołem przenośnych bibliotek oraz narzędzi programistycznych napisanych w C++ do tworzenia aplikacji desktopowych, zagnieżdżonych, a także mobilnych. Wspiera on takie systemy jak: Linux, OS X, Widows, xWorks, QNX, Android, iOS, BlackBerry, Sailfish OS i dzięki czemu jest nadzwyczaj uniwersalnym narzędziem kompatybilnym z następującymi językami programowania: C++, QML (QT Modeling Language), Python, Ring, Go, Rust, PHP i Java. **? ?** QT zapewnia listę dodatkowych możliwości rozszerzających C++. Są to między innymi:

- mechanizmy pozwalające na komunikację między obiektami, zwane sygnałami i otworami (slots);
- wyjątkowa możliwość edycji wyglądu i responsywności obiektów ;

- swobodna możliwość edycji zachowań w przypadku różnego rodzaju zdarzeń ;
- kontekstowe tłumaczenie stringów do internacjonalizacji;
- hierarchiczne i responsywne drzewa obiektowe, organizujące strukturę obiektów;
- automatyczna zmiana wartości wskaźników na 0 w przypadku zniszczenia obiektu, w przeciwieństwie do wskaźników w C++, które stając się zawieszonymi (dangling pointers) **?**;

1.2 Architektura systemu

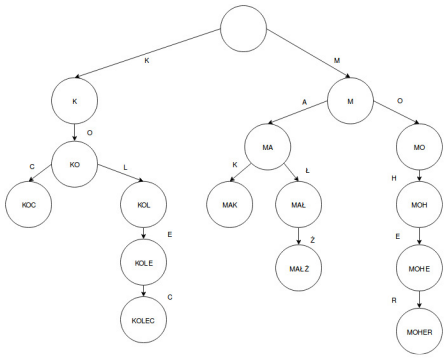
System składa się z warstwy aplikacji oraz warstwy transportowej. Komunikacja zrealizowana jest za pomocą broadcastu przy pomocy protokołu UDP (user data protocol).

1.2.1 Architektura aplikacji

Aplikację zrealizowano w wyżej wymienionym języku C++ z wykorzystaniem wielu bibliotek QT np. QWidget, QString itd.. Do funkcji autouzupełniania tekstu posłużono się algorytmem wykorzystującym skompresowane drzewo trie.

1.2.1.1 Zkompresowane Drzewo Trie

Drzewo Trie jest drzewem służącym docelowo do sortowania wartości danych tesktowych, w którym każdemu węzłowi przypisany jest wspólny prefix(fragment klucza). Wartości tekstowe zapisywane są w liściach drzewa. (**?**) Do drzewa wczytywane zostają wszystkie słowa z danego mu słownika, a następnie każde rozkłada się na litery i rozmieszcza w drzewie, tak by czytając znaki od korzenia do liścia tworzyły pożądany ciąg. Biorą na przykład słownik składający się z następujących słów: mak, róża, kolec, małż, koc, moher otrzymujemy następujące drzewo 1.1. W ten sposób poszukując słów zaczynających się na ”ko” w



Rys. 1.1 – Przykładowe drzewo typu Trie.

bardzo prosty sposób możemy określić, że zaliczają się do nich koc oraz kolec. Drzewo to w znaczący sposób skraca czas przeszukiwania dużych słowników (w wypadku tego projektu 3639970 wyrazów) i umożliwia np. autouzupełnianie albo korektę słów bieżąco wpisywanych przez użytkownika.

Zastosowany algorytm wykorzystuje fakt występowania wspólnych węzłów i zapamiętuje jedynie numer ostatniego węzła związanego z wyszukiwanym słowem np. dla frazy "ko" - węzeł nr. 2, a następnie pobiera wartości wszystkich dzieci tego węzła, zespala je i tworzy wszystkie możliwe końcówki (tu "c" oraz "lec"), które w połączeniu z szukaną frazą dają "autouzupełniane" słowa.

1.2.2 Komunikacja aplikacji

System umożliwia komunikację poprzez broadcast z innymi użytkownikami oraz poprzez specjalne API Google z przeglądarką internetową.

1.2.2.1 Broadcast

Broadcast jest metodą komunikacji (rozsyłania danych) między jednym nadawcą, a wieloma odbiorcami w tym samym czasie. Jest to dyfuzyjny, jednokierunkowy tryb transmisji, charakterystyczny dla sieci LAN. Protokołem regulującym dany ruch sieciowy jest w tym wypadku UDP (User Datagram Protocol). Jest to bezpołączeniowy protokół, który pozwala aplikacjom dobudować własne - niezbędne do poprawnego działania, protokoły. Zapewnia on aplikacji możliwość wysyłania enkapsulowanych datagramów IP bez potrzeby wytworzenia połączenia, co jest jednocześnie dużo szybszą metodą komunikacji, niż tryb synchroniczny. Nie dba on o to, czy wysłane ramki dotrą do odbiorcy w całości, co jest typowe dla komunikacji asynchronicznej. UDP transmituje segmenty złożone z 8-bajtowego nagłówka oraz fragment przesyłanych danych, będący formą wiadomości. UDP zapewnia informację o portach źródłowych i docelowych. Port źródłowy jest przydatny w momencie, gdy urządzenie odbiorcy zechce odesłać odpowiedź na otrzymany segment. Rozmiar ramki wynosi od 8-65 515 bajtów. **?**

1.2.2.2 Google Custom Search

Google Custom Search umożliwia stworzenie własnej wyszukiwarki pozwalającej na przeszukiwanie zarówno stron internetowych jak i obrazów. Możliwe jest zawężenie i personalizacja wyników wyszukiwania np. do wyników pochodzących z konkretnej strony, lub zawierających sprecyzowaną frazę. Google Seachr Engine występuje w dwóch wersjach: darmowej - Custom Search Engine oraz płatnej - Google Site Search. Do potrzeb projektowych wystraczająca jest wersja darmowa, która pozwala na korzystanie z dodatkowych API umożliwiających zwrócenie wyników wyszukiwania w postaci pliku XML czy też JSON. API te upraszczają komunikację aplikacji z przeglądarką do zapytań RESTowych typu get w celu otrzymania uporządkowanej struktury danych. **?**

1.3 Wymagania funkcjonalne

Po przeanalizowaniu problemu klawiatury ekranowej oraz potrzeb osób chorych na ALS stwierdzono następujące wymanagania funkcjonalne:

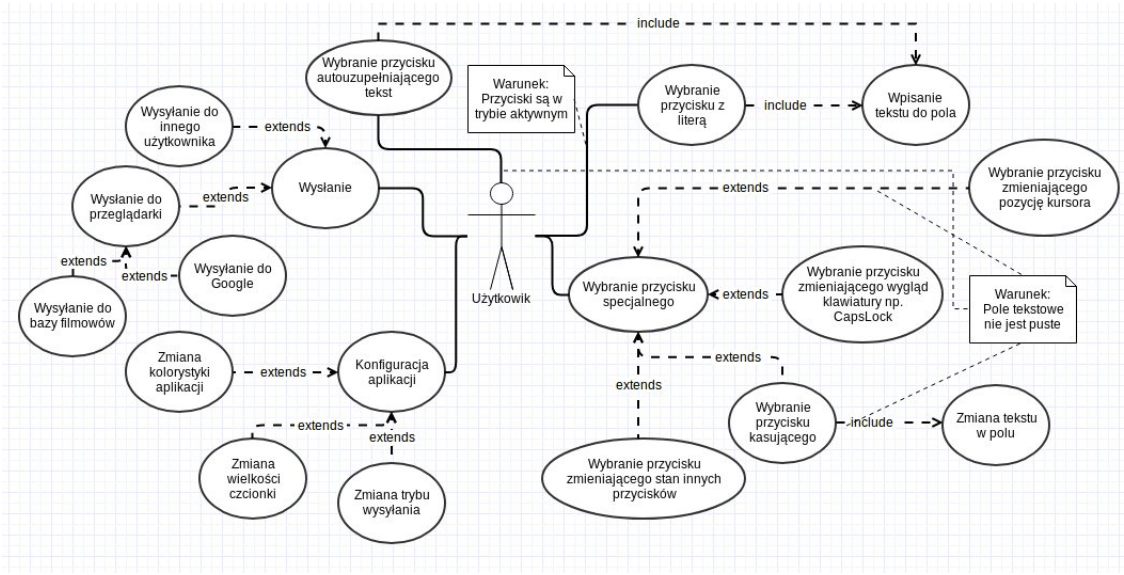
- Klawiatura ekranowa reagująca na sygnał wejściowy będący współrzędnymi punktu fiksacji, które określane są z dokładnością jednego stopnia. Zakładając, że odległość użytkownika od ekranu to 60 cm, wyliczono iż dokładność wyznaczania pozycji to obszar ok. 3,2 cm powierzchnii ekranowej. Dla komputera o ekranie 14 cal (monitor testowy), gdzie rozmiar pojedynczego piksela to ok. 0,2269 mm - to, w przybliżeniu, 141px dla każdego przycisku.
- Możliwość autouzupełniania słów za pomocą słownika języka polskiego.
- Możliwość wprowadzenia przycisków w stan nieaktywny.
- Wykorzystywanie polskich znaków w tekście.
- Wysyłanie wiadomości za pomocą broadcastu.
- Wyszukiwanie stron w przeglądarce Google.
- Możliwość konfigurowania wyglądu aplikacji - różne stopnie kontrastu oraz kolorystyki, a także rozmiaru czcionki tekstu wpisywanego.

Szczegółowe przypadki użycia klawiatry zobrazowano w rozdziale "Przypadki użycia" 1.4.

1.4 Przypadki użycia

Na zamieszczonym rysunku 1.2 przedstawiono diagram przypadków użycia przedstawiający możliwe interakcje użytkownika z aplikacją.

Aby rozpocząć pracę z klawiaturą należy odblokować pisanie znaków wybierając przycisk *start*. Gówną funkcjonalnością, a zarazem najczęściej zachodzącym przypadkiem użycia będzie wybór przycisku z literą w celu wypisania tekstu na ekranie. Użytkownik ma do wyboru klawiaturę podstawową (z 28 znakami oraz cyframi z zakresu 0-9), 2 tablice przycisków ze znakami specjalnymi oraz jedną poświęconą jedynie polskim znakom. Każdy z przycisków reaguje na zmianę położenia punktu fiksacji - w momencie, w którym wykryte zostanie położenie nad przyciskiem naliczany zostaje czas, którego narastanie wizualizuje się za pomocą paska postępu. Gdy 100% postępu zostanie osiągnięte dana litera pojawia się na ekranie w przeznaczonym obszarze i w określonej, za pomocą kursora, pozycji w tekście. W przypadku, jeśli użytkownik zechce zmienić wielkość liter może wybrać między opcją *CapsLock* (stałym powiększeniem liter wpisywanych), bądź *Shift* (zwiększającym kolejną literę dodaną do tekstu, następnie zmieniając wielkość liter na małe). W obu wypadkach następuję zmiana wyglądu całej klawiatury. Po tekście można poruszać się za pomocą strzałek - ruch w lewo (do początku tekstu) i prawo (na koniec tekstu) o jeden znak, lub specjalnych przycisków *home* oraz *end*, które przenoszą kursor odpowiednio



Rys. 1.2 – Wykres przypadków użycia stworzony za pomocą strony Gliffy.com

na początek i koniec tekstu. Przyciski *Enter* oraz *Spacja* traktowane są jako zwykłe znaki. Fiksacja na przycisku Backspace usuwa ostatni wpisany znak, a na przycisku *czyść* wszystko co znajduje się w polu tekstowym. Wyróżnia się też innego rodzaju przyciski specjalne, zmieniające wygląd klawiatury. Są to przyciski do znaków specjalnych oraz polskich liter. Pierwszy z wymienionych w pierwszym kroku prezentuje wszystkie znaki specjalne powszechnego użytku np. wykrzyknik, cudzysłów, średnik, a przy powtórnym wciśnięciu na klawiaturze pojawiają się popularne emotikony, które znalazły tam swoje miejsce, ze względu na możliwość wysyłania wiadomości do innych użytkowników i mają za zadanie ułatwienie reprezentacji emocjonalego przekazu wiadomości. W przypadku, gdy użytkownik wpisał już jakiś fragment tekstu może posłużyć się jednym z przycisków podpowiedzi, które działają na zasadzie autouzupełniania tekstu na podstawie słów znalezionych w słowniku. Aby zmienić wygląd aplikacji lub tryb wysyłania należy skorzytać z przycisku *menu*. To pozwoli na wybranie schematu kolorystycznego aplikacji oraz wilekości czcionki. Użytkownik może wybrać również jeden z trzech trybów rozsyłania zapisanego przez siebie tekstu - są to: *Google Search*, *Filmweb* oraz *Broadcast* (w celu czatu).

1.5 Propozycja rozwiązania

Jedną z możliwych propozycji rozwiązania, którą postanowiono zrealizować jest oprogramowanie komputerowe, które wyglądem przypominać powinno klawiaturę ekranową, czułą na pozycję kursora, z wbudowanym edytorem tekstu. Klawiatura ta powinna realizować żądanie użytkownika w zakresie wyżej wspomnianych wymagań funkcjonalnych tj. pisanie tekstu, wyszukiwanie treści w internecie oraz komunikacją z innymi użytkownikami poprzez broadcast. Co więcej, interfejs użytkownika powinien pozwolić na dostosowanie wyglądu do

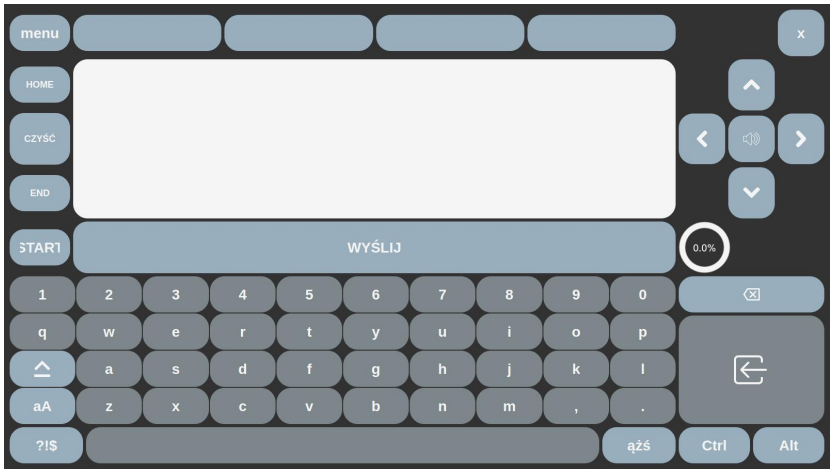
indywidualnych potrzeb. W wynniku pierwszej analizy powstał przedstawiony w podrodziale 1.5.1 prototyp oprogramowania.

1.5.1 Prototyp interfejsu oprogramownia

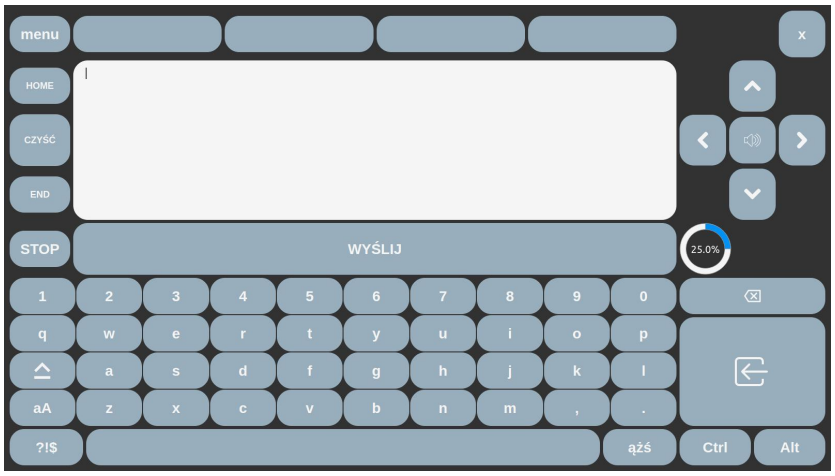
Tworząc prototy oprogramowania zwrócono szczególna uwagę na funkcjonalność interfejsu. Aplikacja otwiera się w trybie pełnoekranowym, tak by otrzymać maksymalny dostępny rozmiar elementów zawartych w interfejsie. Zamieszczono w nim opisane powyżej przyciski niezbędne do poprawnego działania programu. Użytkownik informowany jest o pozostałym czasie do wyboru przycisku poprzez zmieniający się pasek postępu.

Kolorystyka aplikacji została dobrana tak, by chory na stwardnienie boczne rozsiane mógl korzystać z niej nie męcząc oczu podczas długiego korzystania. Połączenie kolorów białego i niebieskiego daje wystaczający kontrast dla ludzkiego oka, by przy niewielkim wysiłku odczytać zapisany tekst. Nie można było zastosować interfejsów czarno-białych o najwyższym poziomie kontrastu ze względu na to, iż obłe kształty przycisków w tych kolorach powodowały powstawanie złudzenia optycznego, które uniemożliwiało pracę na klawiaturze.

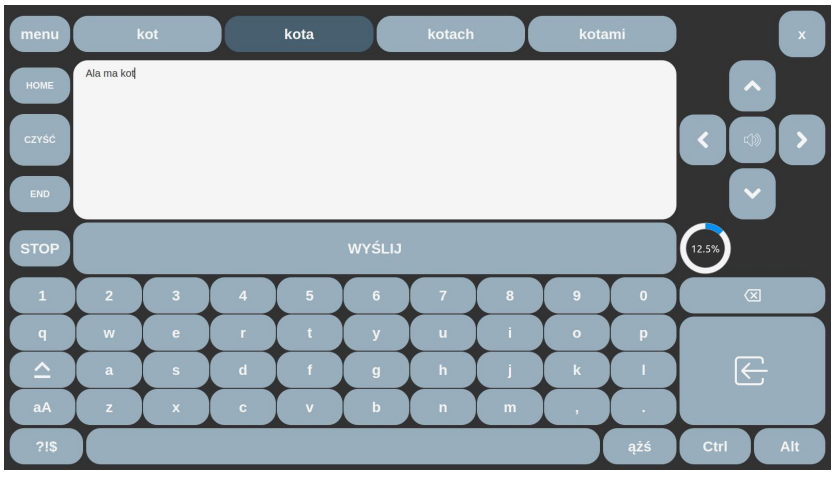
Pierwszym widokiem prezentowanym użytkownikowi jest widok z zablokowaną klawiaturą przedstawiony na rysunku 1.3. Zadbano o to, by przyciski w trybie "wstrzymania" przedstawione były w innym kolorze od przycisków aktywnych. Po użyciu przycisku *START* klawiatura zmienia swój wygląd na jednolity, taki jak na rysunku 1.4. Dalesze dwie grafiki 1.5 oraz 1.6 prezentują wpisywanie tekstu oraz wyświetlanie się możliwych opcji autouzupełnienia tekstu wpisywanego.



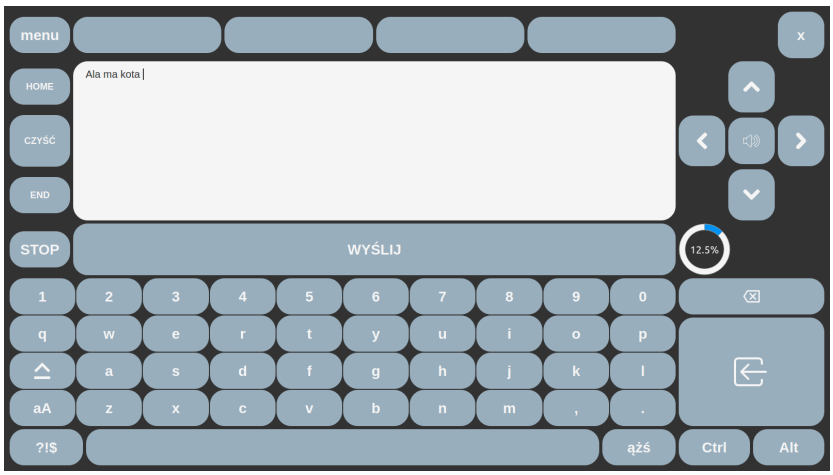
Rys. 1.3 – Początkowy wygląd aplikacji z nieaktywnymi przyciskami.



Rys. 1.4 – Aktywny wygląd aplikacji.



Rys. 1.5 – Przykład wyświetlanych opcji autouzupełniania.



Rys. 1.6 – Tekst i podpowiedzi po wybraniu sugerowanego tekstu.

1.6 Projekt testów

Podczas testowania aplikacji szczególą uwagę należy zwórcić na czas, jaki zajmuje użytkownikowi wprowadzenie tekstu do edytora za pomocą klawiatury ekranowej, przydatność udogodnień typu autouzupełnianie słów oraz subiektywne odczucia osób badanych. W związku z tym zaplanowano następujący przebieg testów:

1. Pomiar czasu wprowadzania tekstu numer 1 do edytora tekstu za pomocą klawiatury komputerowej.
2. Pomiar czasu wprowadzania tekstu numer 2 do edytora tekstu za pomocą klawiatury ekranowej.
3. Pomiar czasu wprowadzania tekstu numer 1 do edytora tekstu za pomocą klawiatury ekranowej.
4. Pomiar czasu wprowadzania tekstu numer 2 do edytora tekstu za pomocą klawiatury komputerowej.
5. Pomiar czasu wprowadzania dowolnego tekstu o określonej tematyce do edytora tekstu za pomocą klawiatury ekranowej.

W celu popranego przebiegu testów należy przyjąć pewne założenia:

- osoby testowane nie będą miały możliwości wcześniejszego zapoznania się z treścią wpisywanych tekstów, ani nie będą znały tematu ostatniego z punktów testu,
- osoby testowane piszą obcym sobie komputerze, by nie znały układu klawiszy komputerowych,
- testujący do wprowadzania danych na klawiaturze ekranowej wykorzystywać będą myszy komputerowej (odrzucamy czynnik braku umiejętności pracy z nowym sprzętem),

- każda z osób testowanych, przejdzie krótkie szkolenie z zakresu działania klawiatury ekranowej,
- w czasie pomiaru czasu zapisywana będzie również ilość wykorzystywanych podpowiedzi tekstu w celu określenia ich skuteczności,
- pod koniec testów z każdym z badanych przeprowadzona będzie rozmowa na temat jego subiektywnych wrażeń podczas pracy z klawiaturą.

1.7 Możliwość dalszego rozwoju

Przewidziano możliość dalszego rozwoju aplikacji wprowadzając do prototypu przyciski, których działania nie obsłużono. Są nimi *Ctrl*, *Alt* oraz *Tekst to Speech*. Dwa pierwsze przewidziane zostały w celu współpracy z programami zewnętrznymi oraz wprowadzenia skrótów klawiszowych (np. Ctrl+A, Ctrl+V, Ctrl+C, Ctrl+X). Należy pamiętać, że użytkownikami klawiatury są często osoby starsze, które przed zachorowaniem miały już styczność z pracą na komputerze. Wprowadzenie znanych im skrótów, czy funkcjonalności może usprawnić ich pracę z komputerem. Bardzo ważnym aspektem oferowanym przez innych producentów oprogramowania dla osób spraliżowanych, bez możliwości komunikacji werbalnej jest TextToSpeech, czyli synteza mowy na postawie tekstu. Tak zsyntetyzowana ścieżka dźwiękowa z wprowadzoną wypowiedzią mogłaby zostać odtworzona poprzez głośniki urządzenia i umożliwić werbalną komunikację z innymi osobami.

1.8 Wymagania pozafunkcjonalne

Podstawowymi wymaganiami pozafunkcjonalnymi są:

- komputer lub urządzenie mobilne z systemem operacyjnym Linux, bądź Windows,
- mysz komputerowa lub specjalne okluary do eye-trackingu,
- dwa monitory do pracy z internetem,
- 4GB pamięci RAM,
- min. 3,5 GB wolnej pamięci,
- monitor o rozdzielczości nie mniejszej niż 14 cal.

1.8.1 Ograniczenia

Ze względu na ograniczoną powierzchnię ekranową nie udało się zrealizować przycisków o minimalnych wymiarach 140px x 140px.

1.9 Podsumowanie

Rozdział ten stanowi analizę techniczną zaplanowanego oprogramowania. Tłumaczy jego działanie oraz przewidziane technologie. Przedstawiono przewidywane wymagania funkcjonalne oraz wynikające z analizy sprzętowej wymagania pozafunkcjonalne. Dokładnie omówiono przypadki użycia aplikacji oraz oferowane funkcjonalności. Poświęcono także podrodział, by omówić ewentualne możliwości rozwoju aplikacji w celu jej udoskonalenia, które nie są celem tego projektu.