Podpisy biometryczne na tablecie i ich porównanie z podpisami na papierze

Raport końcowy

Mikołaj Balcerek (lider zespołu)
Bartosz Hejduk
Mieczysław Krawiarz
Adam Kulczycki
Mikołaj Pabiszczak
Michał Szczepanowski
Dawid Twardowski
Adrianna Załęska

Poznań, wrzesień 2017

Wprowadzenie

Wstęp

Zrealizowany projekt dotyczył zbierania i analizy podpisów biometrycznych. Dynamiczny podpis biometryczny służy weryfikacji tożsamości osoby go składającej. Weryfikacja odbywa się na podstawie porównania cech uśrednionego podpisu wzorcowego (składa się nań zwykle ca. 5 podpisów) z cechami podpisu składanego. Pośród cech, które mogą być brane pod uwagę wyróżnia się m.in.:

- kształt graficzny podpisu, w tym kształt liter;
- właściwości fizyczne sygnatury, rozmiar pociągnięć oraz czas ich złożenia;
- nacisk jaki pióro wywiera na powierzchnię pisania i jego zmienność w czasie;
- ruchy wykonywane piórem podczas pisania - szybkości i przyspieszenia pióra w kierunkach ustalonych osi współrzędnych, jak również liczba oderwań pióra od powierzchni pisania;
- położenie pióra względem płaszczyzny pisania - kąt pisania i jego zmiany w czasie.

Powyższe cechy nie są jednakowe w każdej chwili w czasie składania podpisu, stąd algorytmy weryfikacji analizują zbliżoność rozkładu tychże cech w czasie („górki i dołki występować winny w analogicznej kolejności i wielkości, jednakże być może w nieco różnych odległościach od siebie"). Algorytmy weryfikacji mogą bazować m.in. Ukrytych Modelach Markowa (ang. *Hidden Markov Models*) czy Dynamicznym Marszczeniu Czasu (ang. *Dynamic Time Warping*).

Porównywanie podpisów odbywa się w sposób przybliżony. Efektem porównania jest ocena w postaci punktacji lub procentu określająca podobieństwo złożonych podpisów. Ustalenie progu, decydującego o zaklasyfikowanie podpisu jako prawdziwego, dopuszcza pewien poziom błędu, którego uwzględnienie decyduje o fałszywym odrzuceniu prawdziwego podpisu (ang. *False Rejection*) i o fałszywym przyjęciu nieprawdziwego podpisu (ang. *False Acceptance*).

Zakres projektu

Projekt obejmował stworzenie aplikacji do zbierania i weryfikacji podpisów składanych na urządzeniu *Microsoft Surface* przy użyciu pióra *Microsoft Surface Pen* wraz z opracowaniem algorytmu weryfikacji i stworzeniem bazy danych podpisów (obejmującej podpisy wzorcowe, podpisy prawdziwe i fałszywe), na podstawie której testowano stworzone algorytmy weryfikacji podpisów dynamicznych.

Model fizyczny

Aplikacja

Aplikacja *Podpis Biometryczny* składa się z programu **PodpisBio** oraz wspierającej go bazy danych.

Program

Program **PodpisBio** został zorganizowany w dwa główne podfoldery, *Src* oraz *View*.

Podfolder *View* zawiera implementację graficznego interfejsu użytkownika i jest podzielony na osobne podstrony, takie jak stronę składania podpisów (*SignaturePage.xaml*) lub stronę z danymi statystycznymi (*StatisticPage*).

W podfolderze *Src* znajduje się główna implementacja projektu. Dzieli się ona na subfoldery *Author*, *FinalScore* oraz *Service*. Ostatni z nich zajmuje się łączeniem i aktualizowaniem projektu z lokalną bazą danych podpisów.

W *Author* znajdziemy klasę implementującą Autora (*Author.cs*) oraz jej kontroler. Pozwala to na przypisanie podpisów do różnych autorów i rozróżnianie pomiędzy nimi. Dalej, implementację klasy Sygnatura (*Signature.cs*), przechowującą i zajmującą się obsługą sygnatur. Pomocnicze klasy Punkt oraz Pociągnięcia (*Point.cs*, *Stroke.cs*) dzielą podpis na pomniejsze jego części i są przechowywane jako listy. Klasy *TimeSizeProbe.cs* oraz *Dynamics.cs* badają odpowiednio zależności rozmiaru podpisu do czasu jego złożenia (dla całej sygnatury jak i również dla osobnych pociągnięć) oraz przyspieszenia i szybkości poruszeń pióra *Microsoft Surface Pen*.

Folder *FinalScore* zawiera klasy zajmujące się określaniem finalnego stopnia pewności w autentyczność podpisu oraz metodę DTW (*Dynamic Time Warping*).

Klasy i foldery pomocniczne *Weight*, *FileController*, *RealScreenSizeCalculator* zajmują się odpowiednio obliczaniem wag poszczególnych metod weryfikacji podpisu, zapisywaniem podpisów w formie graficznej i w pliku .csv, oraz obliczaniem rzeczywistego pola podpisu na podstawie informacji o przekątnej ekranu oraz jego rozdzielczości.

Poznańskie Praktyki Badawcze

UAM

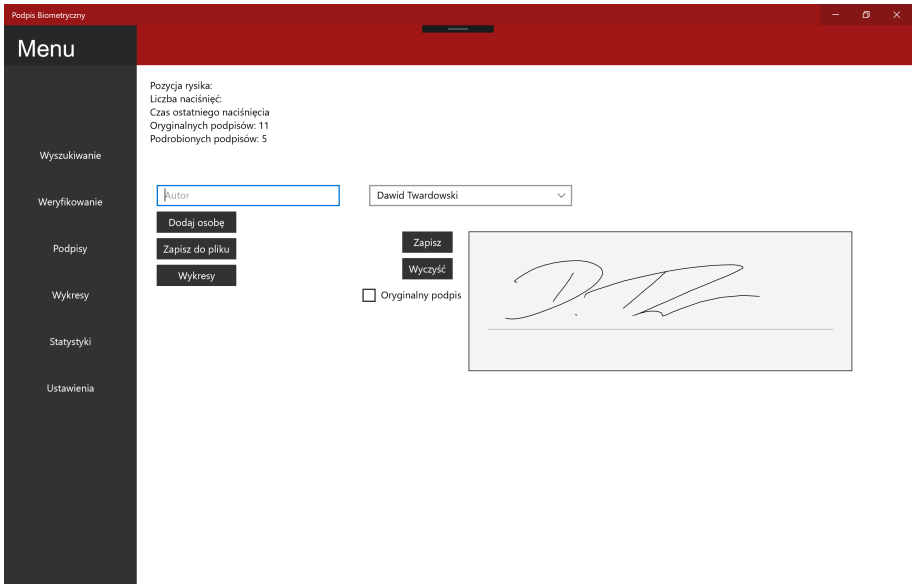
Model logiczny - zrealizowane funkcjonalności

Składanie podpisu

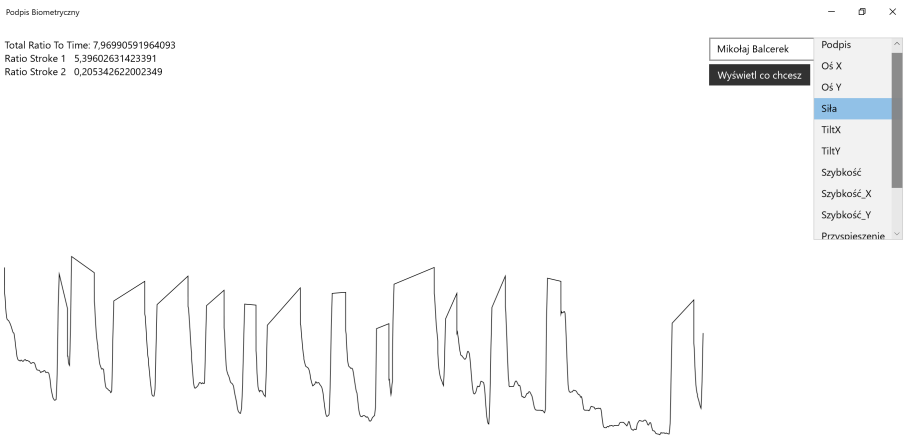
- rejestracja podpisu - informacje zawierające jego fizyczny rozmiar, szybkości, przyspieszenia, siłę naciśnięć rysika oraz czas złożenia podpisu;
- pole składania podpisu dynamicznie zmieniające rozmiar w zależności od przekątnej i rozdzielczości wykorzystywanego ekranu, tak by zachować wymagane pole ok. 110mm x 40mm;
- pole składania podpisu zawierające pomoce (linie wodzące) oraz możliwość wycofania podpisu;
- obsługa gumki znajdującej się na końcu *Microsoft Surface Pen* jako wycofania podpisu;
- zarządzanie listą autorów, synchronizowaną z bazą danych;
- możliwość przyporządkowania podpisów do danego autora;
- określanie prawdziwości lub fałszywości podpisu w trakcie jego dodawania do bazy danych;
- opcjonalna pomoc w tworzeniu fałszywych podpisów w formie nakładki oryginalnego podpisu w polu do pisania;
- synchronizacja bazy autorów i podpisów z lokalną bazą danych.

Pogląd podpisów

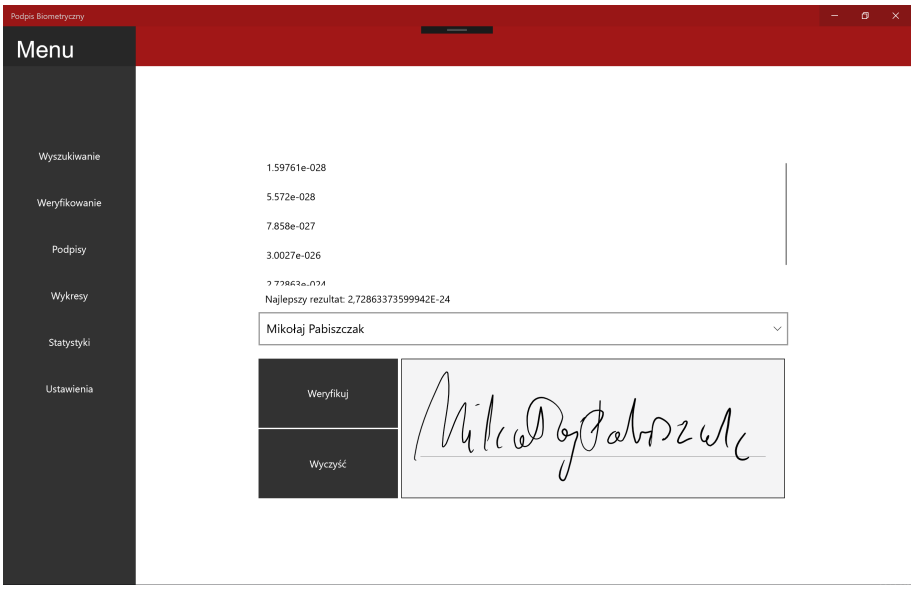
- możliwość wyświetlania podpisów dla wszystkich autorów znajdujących się w bazie danych;



Rysunek 1: Okno podpisu, tutaj można złożyć podpis dla zaznaczonego autora, spróbować go podrobić oraz zapisać do plików



Rysunek 2: Okno poglądu sygnatur z bogatymi opcjami generowania wykresów na podstawie różnych zebranych danych



Rysunek 3: Składanie sygnatur i pogląd wyników autentykacji. Złożony na rysunku fałszywy podpis słusznie otrzymuje bardzo niski wynik pewności na poziomie ok. 0,00.....3

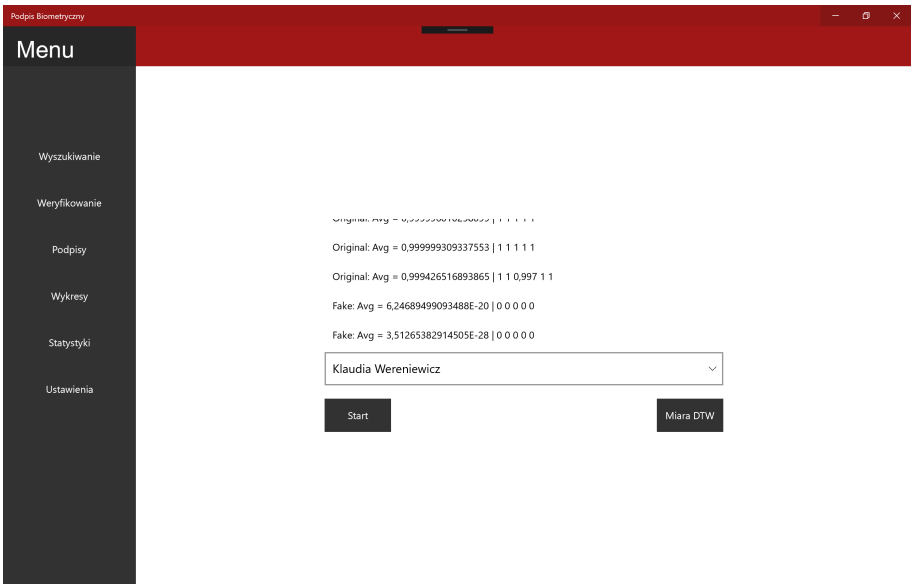
- pogląd metryk TimeSize dla podpisów z osobna;
- pogląd wykresów przyspieszeń, przyspieszeń w osiach, szybkości, szybkości w osiach oraz sił naciśnięć;
- zapisywanie podpisów do plików graficznych .gif;
- zapisywanie podpisów w formie niezmodyfikowanej do plików .csv.

Normalizacja podpisów

- wszystkie złożone podpisy są skalowane do stałych rozmiarów, tak by mimo różnic w rozmiarze rzeczywistym były one łatwe do porównania;
- podpisy po złożeniu są odpowiednio centrowane i przesuwane tak, by wszystkie sygnatury zaczynały się w jednym punkcie i nie zawierały luk.

Możliwości weryfikacji podpisu

- weryfikowanie podpisów na podstawie stosunku rozmiaru całego podpisu do czasu jego złożenia (rzeczywistego czasu wodzenia rysika po ekranie);



Rysunek 4: Pogląd wyników weryfikacji dla wszystkich podpisów złożonych w imieniu danej osoby. Szczególnie warto zwrócić uwagę na wysoką pewność podpisów oryginalnych na poziomie 0,999.. w porównaniu z niską autentycznością sygnatur fałszywych (0,00....6)

- szybkie odrzucanie podpisów zbyt krótkich lub znacząco za długich w obydwu osiach;
- sprawdzanie autentyczności sygnatur korzystając ze stosunku rozmiarów (wykorzystanego wirtualnego "atramentu") osobnych pociągnięć do czasu rzeczywistego ich złożenia;
- określanie autentyczności podpisu na podstawie funkcji zmian siły nacisku (czy siła nacisku maleje, jest stała czy rośnie w czasie);
- badanie podpisów metodą Dynamic Time Warping, biorącą pod uwagę współrzędne, przyspieszenia, szybkości oraz siły nacisku podpisu;
- możliwość przypisywania wag dla poszczególnych metod badania autentyczności podpisu;
- wczesna wersja przypisywania indywidualnej ważności poszczególnych metod weryfikacji dla każdego użytkownika z osobna;

Badanie skuteczności programu

- osobna strona pozwalająca na szybkie testowanie metod weryfikacji wobec wybranej osoby;

- finalny wynik weryfikacji jako liczba 0-1 określająca prawdopodobieństwo prawdziwości składającego podpis;
- nauka algorytmów wzorca podpisu danej osoby na podstawie tylko 3-5 podpisów
- określanie prawdopodobieństwa autentyczności zarówno prawdziwych jak i fałszywych podpisów na podstawie wybranych pięciu wzorcowych podpisów.
- strona ze wstępnymi informacjami statystycznymi;
- badanie skuteczności wszystkich metod wobec wiadomego statusu oryginalny/fałszywy podpisu dla danego autora;
- wysoka skuteczność i dokładność metody DTW, słusznie określająca fałszywość sygnatur dla większości sygnatur, z możliwością określenia wyniku do 30+ miejsc po przecinku

Opis środowiska

Użyte technologie

Program *Podpis Biometryczny* został wykonany w języku programowania **C#**, wykorzystując technologię *Universal Windows Platform*. Zastosowanie tej bazy ułatwia dostęp do informacji o pozycji *Microsoft Surface Pen* oraz ogólnym wykorzystaniu ekranu dotykowego. Aplikacje *Universal Windows Platform* w łatwy sposób dostosowują się do pracy na telefonach, tabletach oraz komputerach osobistych. Wymagają jednak korzystania wyłącznie z platformy *Windows 10*.

Serwer z bazą danych oparty jest na technologii **.NET Framework** w wersji 4.6.1. W bazie przechowywane są podpisy oraz ewentualne parametry, których obliczanie każdorazowo byłoby nieefektywne.

Algorytmy weryfikacji podpisów

Współczynniki bezpieczeństwa w biometrii

W biometrii porównuje się cechy przynajmniej dwóch wzorców cech biometrycznych - pierwszy pobierany jest podczas rejestracji użytkownika w systemie, drugi zaś na bieżąco podczas weryfikacji. O ile cechy biometryczne pozostają niezmienne, to wzorce tych samych cech biometrycznych pobierane w różnym czasie są różne. Stąd w przypadku, gdy porównywane wzorce są identyczne można stwierdzić, iż ma miejsce próba nieuprawnionego uwierzytelnienia (oszustwa). Weryfikacja polega na porównywaniu podobieństwa dwóch wzorców. Wynikiem porównywania jest zazwyczaj punktacja lub wyrażona w procentach zgodność wzorców. Przyjęcie odpowiedniego progu (w punktach lub procentach) umożliwia zaklasyfikowanie składanego wzorca jako prawdziwego lub fałszywego; ponadto próg taki można regulować zmieniając wiarygodność uwierzytelniania.

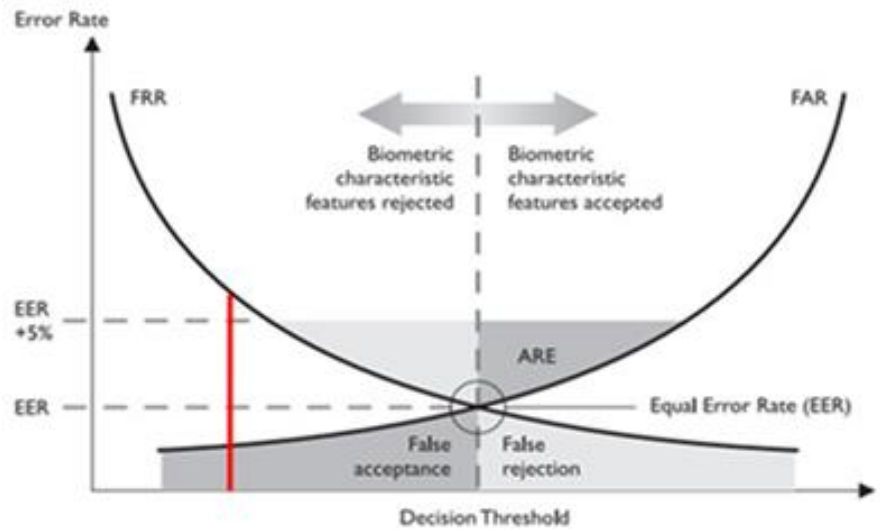
Miarami wiarygodności uwierzytelniania są współczynniki:

- fałszywej akceptacji osoby nieuprawnionej (ang. FAR - *False Acceptance Rate*);
- fałszywego odrzucenia osoby uprawnionej (ang. FRR - ang. *False Rejection Rate*);
- równowagi błędów (ang. EER - ang. *Equal Error Rate*).

Fałszywa akceptacja zachodzi, gdy algorytm klasyfikuje podpis osoby nieuprawnionej za prawdziwy; fałszywe odrzucenie - gdy podpis osoby uprawnionej zostaje zaklasyfikowany jako fałszywy. W praktyce jako parametr porównywania skuteczności różnych metod weryfikacji stosuje się EER - im niższy, tym metoda jest lepsza.

Współczynniki te oblicza się następująco:

$$FAR = \frac{liczba fałszywie zaakceptowanych}{liczba prob weryfikacji}$$



Rysunek 5: Graficzne przedstawienie FAR, FRR, ERR. (Źródło: ISC2, International Information Systems Security Certification Consortium)

$$FRR = \frac{liczba fałszywie odrzuconych}{liczba prob weryfikacji}$$

Zaimplementowany algorytm

Algorytm na wejściu przyjmuje podpis do weryfikacji, listę podpisów wzorcowych danego autora (vide algorytm ustalania wag) oraz wagi dla danego autora. Na wyjściu zwraca informację o tym, czy podpis przeszedł weryfikację.

1. Dla każdego z podpisów wzorcowych wykonuje się kroki 2-4.
2. Dla każdego ustalonego parametru oblicza się podobieństwo tego parametru pomiędzy podpisem do weryfikacji, a podpisem wzorcowym.
   W przypadku DTW wykorzystuje się również próg DTW dla danego autora do obliczenia podobieństwa.
3. Wyniki z kroku poprzedniego mnoży się przez wagę odpowiadającą danemu parametrowi.
4. Sumuje się wyniki dla poszczególnych parametrów.
5. Z otrzymanej listy wybiera się maksimum, które na podstawie określonej metryki określa czy podpis przeszedł weryfikację.

Algorytm ustalania wag

Algorytm na wejściu przyjmuje listę podpisów oryginalnych. Na wyjściu zwraca wagi dla ustalonych parametrów oraz próg dla DTW.

1. Bierze się pierwsze **x** podpisów jako podstawę do wyliczenia wag (**x** ustalone - w naszym przypadku 5).
   Jeżeli **x** jest większe od liczby otrzymanych podpisów to wszystkie traktuje się jako podstawę.

2. Dla ustalonych parametrów oblicza się ich podobieństwo (przedział od 0 do 1) na podstawie podpisów uznanych za bazowe.
   (1 = (odchylenie standardowe / średnia))

3. Przy obliczaniu podobieństwa dla DTW zapisuje się również próg (= średnia) dla danego autora.

4. Wyniki z kroku 2 dla ważniejszych parametrów mnoży się ponadto przez stałą (ustaloną bądź wyliczoną).

5. Wagą dla danego parametru jest wynik podzielony przez sumę wyników dla wszystkich parametrów.

Czego nie udało się zrealizować

Pośród rzeczy, których nie udało nam się (w całości) zrealizować (z różnych powodów) są:

- funkcjonalność dokładnego odrzucania podpisów, które wykroczyły poza pole wprowadzania. Ten feature został zrealizowany częściowo.;
- pobieranie formularza od IC Solutions i wykrywania pola wprowadzania podpisu na skanie kartki (np. wniosku o założenie konta w banku). Potrzeba na tą możliwość została zasygnalizowana na późnym etapie projektu ;
- zbieranie informacji o kątach pisania;
- dopracowanie algorytmów weryfikacji i stworzenie algorytmów opartych na innych metodach (np. Ukrytych Modelach Markowa, probabilistyczne DTW);
- resampling szeregów czasowych;
- wyznaczenie stałego progu dla akceptacji podpisu (dla każdego autora wyznaczany jest indywidualny próg na podstawie 5 podpisów wzorcowych);
- stworzenie statystyk dla różnych algorytmów, które pomogłyby w ich porównywaniu.

Kod źródłowy oraz uruchamianie aplikacji

Wersje programu

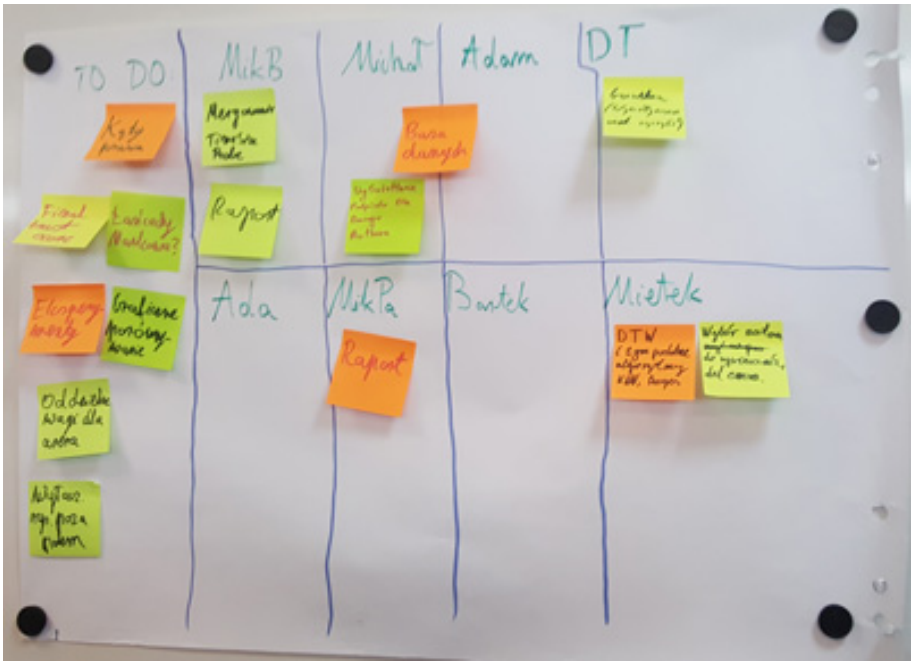
Najnowszą przetestowaną wersję programu można ściągnąć z https://github.com/MikolajBalcerek/PodpisBio/r

Kod źródłowy i dane

Kod źródłowy aplikacji znajduje się na repozytorium pod adresem:
https://github.com/MikolajBalcerek/PodpisBio.

Uruchomianie aplikacji

Do uruchomienia programu wymagana jest praca na platformie *Windows 10 Creators Update* lub wyższej, z zainstalowanym IDE *Microsoft Visual Studio* z modułami *Universal Windows Apps* i obsługą języka C#. Przy pierwszej kompilacji przydatny też jest dostęp do Internetu do automatycznego ściągania pozostałych bibliotek przez środowisko.
W folderze *Program* znajdują się dwa projekty *Visual Studio*, *PodpisBio* oraz *RestService* Obydwa są wymagane do uruchomienia projektu.
Przed uruchomieniem projektu należy z głównego katalogu *Baza Danych* rozpakować znajdujące w paczce *.zip* pliki *.mdf* oraz *.ldf* i przekopiować je do ścieżki *RestService* *App_Data* by uniknąć błędów pobierania wpisów z bazy danych.



Rysunek 6: Dzienny podział pracy w zespole

Bibliografia

1. https://github.com/MikolajBalcerek/PodpisBio
2. Putz-Leszczyńska J., „Biometryczna identyfikacja tożsamości. Biometria podpisu odręcznego"
3. Plucińska M., „Krótki przegląd technik biometrycznych do rozpoznawania osób"
4. Al-Hmouz R., Pedrycz W., Daqrouq K., Morfeq A., Al-Hmouz A., „Quantifying dynamic time warping distance using probabilistic model in verication of dynamic signatures"
5. Llimona Torras Q., „Dynamic Time Warping"
6. Woszczyński T., Marcinkowski Z., Sudoł M. (red.), „Raport biometryczny 2.0. Bankowość biometryczna"
7. Saeed K., Adamski M., „Klasyfikacja podpisu offline z wykorzystaniem metody DTW"
8. Kudłacik P., Porwik P., „A new approach to signature recognition using the fuzzy method"
9. Goc M., „Badania podpisów w kryminalistycznej analizie pismoznawczej - wybrane zagadnienia metodyczne"
10. Hamornik J., „Signature-based User Authentication"
11. Porwik P., Para T., „Some Handwritten Signature Parameters in Biometric Recognition Process"
12. Zwiernik P., „Wstęp do ukrytych modeli Markowa i metody Bauma-Welcha"
13. https://www.tractica.com/biometrics/in-biometrics-which-error-rate-matters/
14. http://www.biometria.sk/en/principles-of-biometrics.html
15. https://www.quora.com/How-can-I-understand-the-EER-Equal-Error-Rate-and-why-we-use-it

16. https://math.stackexchange.com/questions/1292231/what-is-a-decision-threshold-and-how-does-it-apply-to-a-statistical-power