
POLITECHNIKA RZESZOWSKA

im. Ignacego Łukasiewicza

wydział matematyki i fizyki stosowanej

Adrian Jakubowski


# Aplikacja do nauki fiszek

Projekt z przedmiotu

Usługi sieciowe w biznesie

Kierunek studiów:

Inżynieria i Analiza Danych, III rok


Rzeszów 2024

# Spis treści

# 1 Temat projektu

Tematem projektu z przedmiotu "Usługi sieciowe w biznesie" było utworzenie nowego rozwiązania, bądź przetestowanie istniejącego już oprogramowania w zakresie szeroko pojętego biznesu. Z powodu szerokiego wachlarza możliwości, jakie oferował nam ten projekt, mogliśmy się zdecydować na pracę mając do wyboru utworzenie/przetestowanie narzędzia, którego zadaniem jest np. optymalizacja pracy, poprawa wydajności, redukowanie kosztów czy poprawa wiedzy oraz doświadczenia pracownika.

Szukając odpowiedniego tematu projektu, zdecydowałem się na stworzenie czegoś z czego sam będę miał możliwość oraz będę chciał korzystać w życiu codziennym. Z tego względu, zainspirowany portalem Quizlet, postanowiłem utworzyć prostą aplikację okienkową do nauki "wirtualnych fiszek", czyli komputerowych odpowiedników małych karteczek, skutecznych przy nauce m.in. słów oraz zwrotów w językach obcych czy definicji i pojęć z innych dziedzin.

# 2 Cel i zakres pracy

## 2.1 Cel projektu

Celem projektu, zgodnie z ustalonym tematem, było utworzenie aplikacji do nauki fiszek, dzięki którym użytkownik może uczyć się pojęć oraz definicji, analizować swoje wyniki, a następnie sprawdzać swoją wiedzę.

Głównym zadaniem takiej aplikacji jest możliwość tworzenia własnych zestawów fiszek, a następnie ich nauka, poprzez interaktywne "odwracanie" fiszki. Powodem, dla którego użytkownik chcę korzystać z takiego narzędzia jest poprawa swojej wiedzy, więc konieczne było również zaimplementowanie możliwości sprawdzenia swojej wiedzy, poprzez opcję zaznacza, które pojęcia są już znane dla użytkownika, a które należy powtórzyć. Ponadto, zdecydowałem się na utworzenie także dodatkowej opcji w mojej aplikacji, a mianowicie quizu, który losuje pytanie spośród fiszek w wybranym zestawie oraz losuje także 4 odpowiedzi z czego jedna jest prawidłowa. W taki sposób, gracz może szlifować swoją wiedzę.

## 2.2 Metodologia

Spośród gąszcza narzędzi oraz języków programowania, dzięki którym mógłbym utworzyć moją aplikację, zdecydowałem się na język Python, a dokładnie na bibliotekę Tkinter, która pozwala na tworzenie prostych interfejsów graficznych. Umożliwia tworzenie okien, przycisków, etykiet i innych elementów interfejsu. Działanie w powyższej bibliotece opiera się głównie na dodawaniu i usuwaniu odpowiednich elementów okienka, przez co użytkownik ma wrażenie jakby korzystał z różnych podstron na stronie internetowej. Jeśli chodzi o tworzenie zestawów fiszek oraz ich przechowywanie, do tego celu zostały wykorzystane proste, niewielkiego rozmiaru słownikowe bazy danych JSON.


Rysunek 1: Biblioteka Tkinter

Zdecydowałem się na pracę w bibliotece Tkinter ze względu na prostotę tego narzędzia, dobre przystosowanie do tego konkretnego zadania oraz na wcześniejszą styczność z tym rozwiązaniem i idące za tym dobre wspomnienia z nim związane.

# 3 Szczegółowy opis projektu

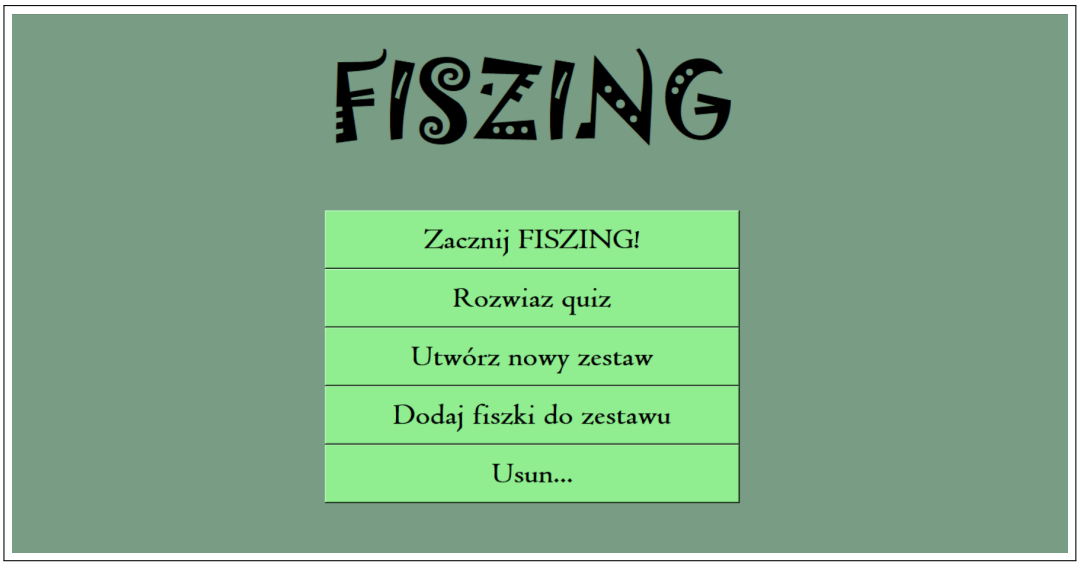
## 3.1 Użyte biblioteki

- **tkinter** - to standardowa biblioteka Pythona do tworzenia graficznych interfejsów użytkownika (GUI)
- **tkinter.messagebox** - moduł messagebox w tkinterze zawiera funkcje do wyświetlania standardowych okien dialogowych, takich jak komunikaty informacyjne, ostrzeżenia i zapytania
- **json** - służy do manipulowania danych w formacie JSON, umożliwiając konwersję między strukturami danych Pythona, a formatem JSON.
- **random** - zapewnia funkcje do generowania losowych liczb oraz wybierania losowych elementów z sekwencji
- **os** - zawiera funkcje do interakcji z systemem operacyjnym, umożliwiając operacje na plikach i katalogach oraz uzyskiwanie informacji o środowisku systemowym

## 3.2 Menu główne

W menu głównym, użytkownik ma do wyboru kilka przycisków odnoszących się do bardziej rozwiniętych elementów aplikacji. Ponadto, możemy zauważyć także nazwę całej aplikacji - FISZING.

Z kwestii czysto technicznych, strona główna jest pierwszym oknem po inicjalizacji aplikacji. Program tworzy instancję klasy **FlashcardsApp** i przekazuje do niej obiekt **root**, jako główny widget biblioteki Tkinter. W funkcji inicjalizacyjnej znajdują się także fragmenty kodu poświęcone konfiguracji okna, związane m.in. z tytułem aplikacji, kolorami czy rozmiarem okna. Ponadto, tworzone są pierwsze przyciski, widoczne na stronie głównej, dzięki którym możemy przejść do innych "stron", a technicznie patrząc, do innych funkcji w kodzie.




Rysunek 2: Ekran główny aplikacji

## 3.3 Dodawanie zestawu fiszek

### 3.3.1 Tworzenie zestawu fiszek

Po kliknięciu przycisku **Utwórz nowy zestaw** użytkownik przechodzi do okna, w którym ma możliwość wpisania nazwy nowego zestawu. Widoczne są również przyciski do zatwierdzenia nowego zestawu oraz powrotu do strony głównej. Aby dodać estetyki swojej aplikacji, utworzyłem także placeholder w polu wprowadzania nazwy nowego zestawu.


Rysunek 3: Ekran wprowadzania nazwy zestawu

Skupiając się na kodzie, po wybraniu przez użytkownika opcji **Utwórz nowy zestaw** w głównym oknie, uruchamiana jest metoda **make new set**, która tworzy pole


Rysunek 4: Wprowadzona nazwa zestawu

do wpisania nazwy zestawu, a także zarządza placeholderem. Tak jak było wspomniane we wstępie, biblioteka Tkinter opiera się na usuwaniu i wyświetlaniu odpowiednich elementów. W tej metodzie usuwane są wszystkie przyciski ze strony głównej, aby kliknięcie powodowało przejście do pozornie "nowej strony".

### 3.3.2 Dodawanie pojęć do nowoutworzonego zestawu

Po zatwierdzeniu nazwy użytkownik otrzymuje możliwość do wpisywania pojęć oraz definicji do nowoutworzonego zestawu. Po potwierdzeniu hasła, powstaje okienko, które wyświetla obecnie zawarte pojęcia i ich definicje w danym zestawie.


Rysunek 5: Ekran wprowadzania pojęć i definicji do zestawu

Potwierdzenie nazwy zestawu uruchamia funkcję **create new set**, dzięki której tworzony jest nowy plik JSON przechowywujący pojęcia i definicje, a także wyświetlane są nowe pole do wpisywania haseł.


Rysunek 6: Wprowadzone pojęcia i definicje wraz z wyświetlaniem obecnych haseł w zestawie

Po zatwierdzeniu hasła kod przechodzi do metody **add flashcard**, która pobiera (get) wpisane pojęcia, zapisuje je do zestawu (zarówno do pliku JSON jak i do okienka, które wyświetla się po zatwierdzenia hasła), a po wszystkim czysci pola inputu hasła. Metoda ta dodatkowo uruchamia metody **save flashcards** oraz **set in frame** odpowiedzialne odpowiednio za dopisywania hasła do pliku JSON oraz tworzenie ramki z pojęciami i wypisywanie do niej tych pojęć.

## 3.4 Dodawanie pojęć do istniejącego zestawu

Aby dodać pojęcia do zestawu, który został stworzony wcześniej, użytkownik korzysta z przycisku **Dodaj fiszki do zestawu** na stronie głównej. Następnie wyświetla się okno wyboru zestawu, a po podjęciu decyzji, w którym zestawie mają być dodane nowe hasła, wyświetla się okno analogiczne do tego, które pojawiało się po utworzeniu nowego zestawu.

Patrząc pod względem kodu, najpierw program przechodzi do funkcji **see all sets**, która jak sama nazwa wskazuje wyświetla użytkownikowi wszystkie zestawy (przy jednoczesnym usuwaniu odpowiednich przycisków). Za pomocą utworzonych w tej funkcji przycisków użytkownik ma możliwość przejścia do kolejnej funkcji - **show set flashcards**, która jako argument wejściowy przyjmuje nazwę zestawu i pozwala na dodawa-


Rysunek 7: Ekran wyboru zestawu

nie nowych pojęć do danego setu oraz wyświetla wszystkie pojęcia w ramce.

## 3.5 Usuwanie fiszek oraz zestawów fiszek

W takiej aplikacji konieczne jest również udostępnienie możliwości do usuwania zbędnych fiszek z określonych zestawów, jak i pozbywanie się całych setów haseł. Za pomocą przycisku **Usun..** użytkownik przechodzi do wyboru czy ma zamiar usunąć konkretną fiszkę z danego zestawu czy cały zestaw.

Program przechodząc do przycisku **Usun...** uruchamia funkcję **delete screen**, która ukrywa przyciski ze strony głównej oraz tworzy przyciski do wyżej opisanego wyboru.

### 3.5.1 Usuwanie zestawu

Jedną z opcji jest usuwanie całego zestawu fiszek. Użytkownik otrzymuje listę przycisków, dla każdego zestawu. Dodatkowo, utworzone zostały informacyjne message boxy, które zapobiegają przypadkowym usunięciom zestawów.

Po wyborze przycisku **Usun...** uruchamiana jest funkcja **delete screen**. Po tym, użytkownik wybierając opcje **Usun zestaw**, która przechodzi do funkcji **show delete sets**. Dzięki temu, osoba korzystająca z aplikacji ma możliwość wyboru zestawu do usunięcia, a gdy dokona wyboru, wyświetlany jest messagebox z pytaniem upewniającym, że jest to czyn zamierzony. Ponadto usuwany jest plik JSON dla danego pliku. Dzieje się to kolejno za pomocą funkcji **confirm delete set** oraz **delete set**.

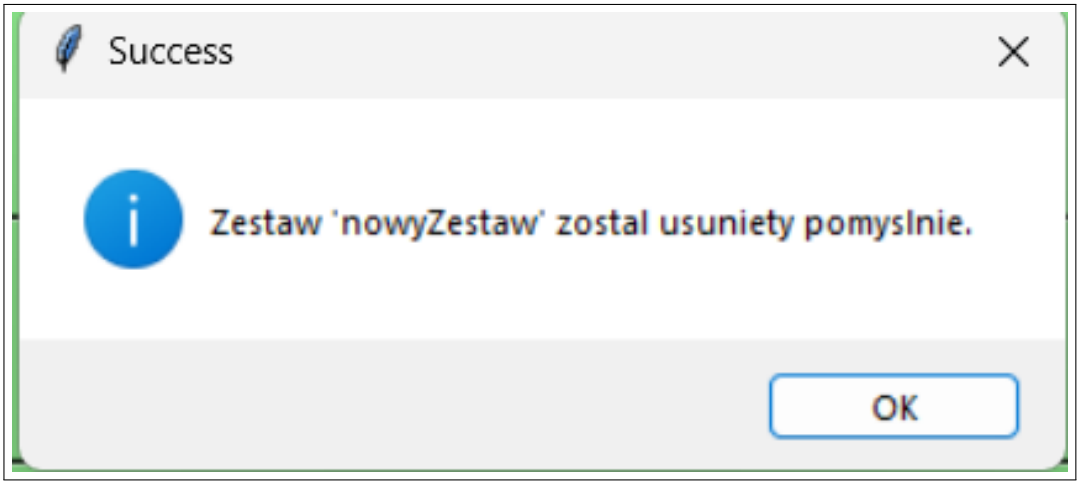

Rysunek 8: Dodawanie pojęcia do istniejacego zestawu wraz z wyświetleniem obecnych haseł w zestawie


Rysunek 9: Wybór opcji usuwania

### 3.5.2 Usuwanie poszczególnych pojęć z zestawu

Poza usunięciem całego zestawu, użytkownik ma możliwość usunięcia poszczególnych fiszek z wybranego zestawu. Podobnie jak w poprzednim przypadku, użytkownik najpierw wybiera zestaw, z którego chce usunąć fiszki, a następnie wyświetlana jest ramka z przyciskami odpowiadającymi konkretnemu pojęciu. Po kliknięciu wybranego hasla, usuwane jest ono z bazy, a ramka z przyciskami jest odświeżana. W tym przypadku, pominięto funkcję messageboxów, gdyż potencjalna przypadkowa utrata pojedynczego hasła w bazie nie jest tak drastyczna. Ponadto, brak okienek dialogowych potwierdzających działanie, przyspiesza proces usuwania większej ilości fiszek.


Rysunek 10: Okno dialogowe dotyczące usuwania zestawu fiszek




Rysunek 11: Komunikat po usunięciu zestawu

Pod względem kodu, początek działania jest analogiczny do czynności usuwania całego zestawu. Zadanie polegające na usuwaniu poszczególnych pojęć zestawu rozpoczyna się w funkcji **delete cards**, gdzie użytkownik wybiera zestaw, z którego chce usunąć fiszki. Następnie w metodzie **show set flashcards delete** wyświetlana jest nazwa zestawu oraz następuje odwołanie do funkcji **delete flashcard in frame**, gdzie tworzona jest ramka wraz z przyciskami, które odpowiadają konkretnym pojęciom. Ostatecznie, usuwane jest pojęcie i definicja z bazy danych za sprawą metody **delete flashcard**.


Rysunek 12: Ekran usuwania poszczególnych pojęć z zestawu

## 3.6 Nauka fiszek

Najważniejszym zadaniem całej aplikacji jest nauka na podstawie "wirtualnych fiszek". Użytkownik dzięki aplikacji ma możliwość uczyć się poprzez swoiste "odwracanie" fiszki, a następnie zaznaczanie czy dane pojęcie zostało przez niego przyswojone czy jednak wymaga powtórki. Osoba ucząca się może w ten sposób dowiedzieć się ile haseł należy powtórzyć, aby osiągnąć stuprocentową wiedzę z danego zakresu.

Podobnie, jak w większości poprzednich funkcjonalności aplikacji, korzystający najpierw ma do wyboru zestaw, z którego chce się uczyć. Po wyborze, wyświetla się główne okno służące do nauki. Zawiera ono nazwę zestawu, licznik zagadnień oraz liczbę pojęć, które użytkownik potrafi zapamiętać oraz ilość pojęć do powtórzenia. Ponadto, na samym środku jest duży przycisk, który służy do "odwracania" pojęcia, a także dwa dodatkowe przyciski, dzięki którym korzystający określa znajomość danego hasła.

Użytkownik przechodzi do kolejnego pojęcia poprzez wybór jednego z przycisków umiem/nie umiem. Iterowany jest także numer pojęcia oraz zwiększana jest liczba jednego z liczników służących do wyświetlania haseł przyswojonych oraz do powtórzenia.

Po dojściu do końca zestawu, wyświetla się krótkie podsumowanie ukazujące liczbę pojęć, które użytkownik potrafi oraz tych, które wymagają dalszej nauki.

Pod względem samego kodu, funkcjonalność związana z nauką okazała się być najbardziej zawiłą z całej aplikacji, poprzez mnogość funkcji obsługujących każdy, czasami bardzo drobny element, taki jak np. odwrócenie fiszki czy zwiększenie licznika pojęć,


Rysunek 13: Ekran nauki fiszek

które użytkownik zna/nie zna.

Główną funkcją jest **show learning frame**, do której program przechodzi po wyborze zestawu (funkcja przyjmuje również wybrany zestaw jako argument). Tam ukrywane są przyciski zestawów, a tworzone są etykiety wyświetlające m.in. licznik pytań czy samo pojęcie. Co więcej, powyższa metoda ukazuje przyciski do "odwracania" fiszki oraz do określenia jej znajomości.

W funkcjach **on know click** i **on dont know click** iterowany jest licznik pojęć odpowiednio znanych i nieznanych przez użytkownika. W tych metodach istnieje także odwołanie do funkcji **next flashcard**, która zmienia pojęcie na głównym przycisku, a w przypadku zakończenia zestawu, wyświetla wynik.

W fragmentach kodu poświęconych nauce fiszek można wyróżnić jeszcze funkcję **flip flashcard**, która jak sama nazwa wskazuje zamienia pojęcie na jego definicje, powodując imitację odwrócenia fiszki.

## 3.7 Weryfikacja wiedzy - rozwiązywanie quizów

Tak jak wspomniano powyżej, głównym celem aplikacji jest możliwość nauki wcześniej zapisanych pojęć, jednakże czym jest skuteczna nauka bez jej weryfikacji. Z tego powodu, stworzono prostego rodzaju quiz, który wyświetla pytanie (pojęcie) i cztery odpowiedzi (definicje - jedną poprawną i trzy losowe błędne z tego samego zestawu).

W celu ciągłej kontroli swojej wiedzy, postanowiono wyświetlać messageboxy po


Rysunek 14: Ekran nauki fiszek z odwróconą fiszką

każdej odpowiedzi. Ukazują one czy odpowiedź jest prawidłowa, a gdy tak nie jest, to użytkownik widzi również poprawną odpowiedź.

Po ostatnim pytaniu w quizie osobie rozwiązującej ukazuje się wynik końcowy w postaci procentowej. Dzięki takiemu rozwiązaniu użytkownik może określić czy udałoby mu się zdać potencjalny test lub po prostu stwierdzić, że posiadana wiedza jest wystarczająca.

Z perspektywy kodu, początek polega na tym samym co w większości przypadków - wypisanie zestawów i wybór tego, którego ma dotyczyć quiz. Dzieje się to w funkcji **solve quiz**, której głównym zadaniem jest wyświetlenie przycisków zestawów, a następnie odniesienie do metody **start quiz** z argumentem będącym nazwą zestawu. Jest to najważniejsza funkcja jeśli chodzi o quizy w aplikacji, ponieważ wywołuje ona metody **prepare questions** i **show next question**, odpowiedzialne kolejno za przygotowanie listy losowych pytań z zestawu oraz zarządzanie wyświetlaniem pytań, odpowiedzi oraz warunkami stopu. Niezwykle przydatna okazała się w tym elemencie funkcja shuffle, która pozwoliła nam losować listę pytań oraz odpowiedzi. Program, po tym jak użytkownik wybrał odpowiedź, przechodził do funkcji **check answer**, w której sprawdzano poprawność odpowiedzi oraz iterowano licznik odpowiedzi prawidłowych.

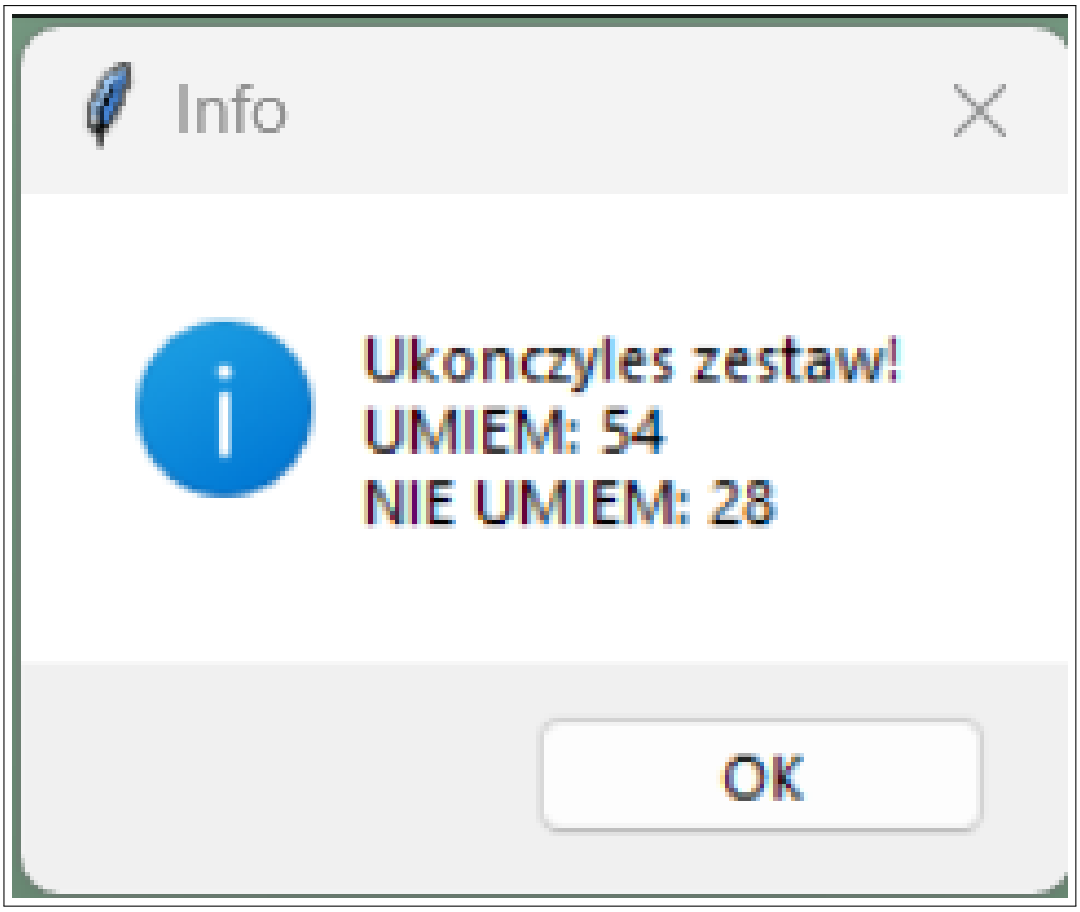

Rysunek 15: Przebieg nauki fiszek

# 4 Potencjalny rozwój projektu

Choć biblioteka Tkinter posiada szerokie spectrum możliwości i jest względnie prosta w zastosowaniu, ma jednak swoje wady. Największą wadą jest trudność w tworzeniu responsywnych okien graficznych. Utworzona aplikacja niestety nie posiada dynamicznych wielkości swoich elementów i jest sztywno przystosowana pod daną rozdzielczość ekranu, co byłoby głównym założeniem dalszego rozwoju projektu. Minusem aplikacji jest także brak scrollbara na całym oknie. Okazało się to bardzo trudne do implementacji mając już wykonane większość projektu. Z racji tego niestety istnieje maksymalna ilość np. przycisków zestawu, które będą widoczne na ekranie. Powyższe aspekty pomogłyby w wydajności i przejrzystości aplikacji mającej obecną ilość funkcjonalności, jednak można poruszyć także kwestie potencjalnych elementów, które ulepszyłyby aplikację w przyszłości.

Bardzo ciekawym pomysłem byłaby umiejętność zapisywania w bazie danych wyników swojej nauki czy quizów, a następnie porównywanie swojego postępu w nauce na przestrzeni czasu. Blblioteka Tkinter posiada możliwości wyświetlania wykresów, więc zdolność wyświetlania wykresów prezentujących ilość nauczonych pojęć przy każdym powtórzeniu quizu/nauki byłoby interesującym aspektem dla wielu użytkowników.

Patrząc pod kątem nauczycieli korzystających z aplikacji, bardzo efektywnym dodatkiem mogłaby być możliwość eksportu utworzonych zestawów haseł czy quizów w postaci plików z rozszerzeniem docx lub pdf. Dzięki takiej funkcjonalności nauczyciele




Rysunek 16: Komunikat po zakończeniu zestawu

mieliby możliwość szybkiego generowania materiałów na swoje zajęcia.

# 5 Wnioski i podsumowanie

Projekt z przedmiotu **Usługi sieciowe w biznesie** dał nam szerokie możliwości rozwoju, również w kwestiach bliższych naszym zainteresowaniom z powodu możliwości swobodnego wyboru tematu. Korzystając z tej możliwości postanowiłem stworzyć aplikację, która służy mi na codzień i która jest dobrym polem do dalszego rozwoju w zakresie języka Python, a w szczególności biblioteki Tkinter.

Aplikacja spełnia swoje założenia, gdyż daje użytkownikowi możliwość tworzenia własnych zestawów fiszek, a następnie nauki z nich. Użytkownicy mogą również wykorzystywać funkcjonalność rozwiązywania quizów w celu weryfikacji nabytej wiedzy.

Pod względem technicznym, dzięki temu projektowi udało mi się pogłębić wiedzę z zakresu języka Python, ze zdecydowanym naciskiem na bibliotekę Tkinter. Był to również projekt wymagający analitycznego myślenia, aby zrozumieć istotę działa-


Rysunek 17: Ekran główny quizu

nia tej aplikacji i dostosować kolejno wyświetlające się i ukrywane przedmioty. Sama implementacja nie była nad wyraz skomplikowana, gdyż istnieje wiele poradników dotyczących tejże biblioteki, a element raz napisany, często powielał się w projekcie. Najtrudniejsze dla mnie okazało się tworzenie okienek ze scrollbarem, przy których ostatecznie poległem próbując tworzyć scrollbar dla całego okna.

Gwoli podsumowania, opisana powyżej aplikacja pozwoliła mi pogłębić wiedzę z języka Python, który jest jednym z kluczowych technologii dla analityka danych. Dzięki projektowi, nie tylko dokonałem postępu jeśli chodzi o moje umiejętności programistyczne czy analityczne, ale także utworzyłem prostą aplikację codziennego użytku, dzięki której mogę nabywać wiedzę z różnych dziedzin.

# 6 Źródła i pomoce


- *www.youtube.com/watch?v=0WafQCaok6g&t=315s*
- *akademiapython.pl/niech-tego-dotkna-tkinter-od-podstaw*
- *www.youtube.com/watch?v=KJhZKnFNNYM*
- *www.youtube.com/watch?v=BckVJoE94Lk&t=362s*


Rysunek 18: Komunikat o poprawnej odpowiedzi w quizie


Rysunek 19: Komunikat o niepoprawnej odpowiedzi w quizie


Rysunek 20: Wyświetlenie wyniku po zakończonym quizie