

UNIwersytet Rzeszowski
Kolegium Nauk



Krzysztof Sendera

Nr albumu: 117869

Kierunek: Informatyka
i Ekonometria

**System do analizy danych „Pynalise” z wykorzystaniem oraz omówieniem
technologii Python**

Praca inżynierska

Praca wykonana pod kierunkiem

.....

Rzeszów, data

System do analizy danych „Pynalise” z wykorzystaniem oraz omówieniem technologii Python

System for data analysis 'Pynalise' using and discussing Python technologies.

1. Wstęp	4
2. Opis funkcjonalności	5
3. Technologie:	5
3.1. Wybór typu aplikacji:	6
3.2. Biblioteki Pythona wykorzystane w projekcie aplikacji desktopowej:	7
3.2.1. Pandas	7
3.2.2. Matplotlib	8
3.2.3. PyQt6	9
3.2.4. scikit-learn	10
3.2.5. reportlab	10
3.2.6. Seaborn	10
3.2.7. JSON	11
3.3. Biblioteki Pythona wykorzystane w stronie internetowej:	11
3.3.1. Django	11
3.3.2. bootstrap4 for django	11
3.4. Środowisko programistyczne	11
3.5. System kontroli wersji	11
4. Modele Matematyczne zaimplementowane w aplikacji	12
4.1. Klasyfikacja – model algorytmu regresji liniowej	12
4.2. Klasyfikacja – model algorytmu drzewa decyzyjnego	12
4.3. Klasteryzacja – model algorytmu k-sąsiadów	12
4.4. Minimum	12
4.5. Maximum	12
4.6. Odchylenie Standardowe	12
4.7. Średnia arytmetyczna	13
4.8. Korelacja	13
5. Diagramy UML	14
5.1. Diagram przypadków użycia:	14
5.2. Diagramy sekwencji:	20
5.2.1. Diagram sekwencji dla Rejestracji nowego użytkownika	20
5.2.2. Diagram sekwencji dla Logowania do systemu	21
5.3. Diagram Stanów Tabeli	21
6. Proces tworzenia aplikacji	22
6.1. Projektowanie User Interface	22
6.1.1. Przegląd dostępnych technologii na rynku.	22
6.1.2. Wybór PyQt6	22

6.1.3. Projektowanie okien do Pyanalise.	23
6.1.4. Konwertowanie plików .ui do Python	26
6.2. Kodowanie komunikacji między oknami.	27
6.3. Kodowanie przycisków odpowiedzialnych za funkcjonalności systemu.	28
6.3.1. Programowanie metod inicjalizujących klasy okien	28
6.3.2. Kodowanie funkcji odpowiedzialnych za obliczenia matematyczne	29
6.3.3. Kodowanie generowania wykresów	31
6.3.4. Kodowanie wczytywania oraz zapisywania plików	33
6.3.5. Kodowanie generowania raportu PDF	35
6.3.6. Kodowanie wyświetlania informacji o przyciskach po najechaniu na nie kursorem myszy	36
6.3.7. Kodowanie stylów, które umożliwiają wybór motywu aplikacji	36
6.4. Podłączenie projektu do bazy danych z użytkownikami	37
6.5. Eksportowanie aplikacji użytkowej.	37
6.6. Kodowanie funkcji odpowiedzialnych za edycję, aktualizację oraz wypisywanie danych.	37
7. Sposób użytkowania aplikacji	38
7.1. Logowanie	38
7.2. Rejestracja	39
7.3. Wczytywanie Danych	39
7.4. Konfiguracja stylu	41
7.5. Obliczanie statystyk	41
7.6. Wyszukiwanie	43
7.7. Generowanie Wykresów	43
7.8. Klasyfikacja	46
7.9. Klastrowanie	47
7.10. Generowanie Pliku Wynikowego PDF	49
7.11. Zapisywanie Danych	49
8. Testowanie Błędów.	50
8.1. Walidacja danych	50
8.2. Funkcje liczące dane liczbowe	50
8.3. Wczytywanie danych w formacie CSV	51
8.4. Wczytywanie nowych danych	51
9. Podsumowanie	51
10. Bibliografia	52

1. Wstęp

Współczesne rozwiązania informatyczne mają ogromny wpływ na funkcjonowanie wielu branż. Jednym z kluczowych aspektów zarządzania firm jest przetwarzanie danych. W ramach projektu inżynierskiego zdecydowałem się opracować nowoczesny System zarządzania i analizy danych, który umożliwi łatwe i wygodne zarządzanie informacjami. Aplikacja stworzona będzie dla osób zainteresowanych atrybutami oraz statystykami, na podstawie analizy których być może będzie można wysnuć kolejne informacje.

Mechanizmy interakcji użytkownika z aplikacją będą różne, w zależności od roli, którą użytkownik pełni w systemie. Gość będzie mógł przeglądać informacje za pomocą prostego interfejsu graficznego. Użytkownik, dodatkowo będzie miał możliwość wprowadzania zmian w danych oraz dodawania nowych informacji, także za pomocą intuicyjnego interfejsu graficznego. Administrator, poza tymi funkcjami, będzie mógł dodatkowo zarządzać użytkownikami, co również będzie możliwe przez interfejs graficzny.

Prezentacja wyników będzie odbywać się za pomocą różnych widoków, w zależności od potrzeb użytkownika. Gość będzie miał dostęp tylko do widoku informacyjnego, który wyświetli mu dane na temat zwierząt, ich zdjęcia i opisy. Użytkownik będzie miał dodatkowo dostęp do widoku edycji, który pozwoli mu na modyfikowanie danych, dodawanie nowych rekordów oraz usuwanie już istniejących. Administrator będzie miał dodatkowo dostęp do widoku zarządzania Użytkownikami, który umożliwi mu zarządzanie kontami, zmianę ich uprawnień oraz usuwanie.

Potencjalne wąskie gardła mogą pojawić się na poziomie serwera, jeśli będzie on nieodpowiednio skonfigurowany i nie będzie w stanie obsłużyć dużej liczby użytkowników jednocześnie. W takiej sytuacji konieczne będzie zastosowanie odpowiedniego sprzętu lub usług chmurowych, aby zoptymalizować działanie systemu. Innym potencjalnym wąskim gardłem może być sam interfejs graficzny, jeśli nie będzie on intuicyjny i łatwy w obsłudze dla użytkowników. W takim przypadku konieczne będzie przeprowadzenie testów UX i wprowadzenie odpowiednich poprawek.

Celem mojej aplikacji jest przedstawienie użytkownikom informacji o różnych danych. Przykładem reprezentowanym w aplikacji będą dane o zwierzętach, które znajdują się w Zoo w Nottingham. System umożliwi użytkownikom dostęp do różnych informacji, takich jak gatunek zwierzęcia, jego opis, atrybuty, itp. Dzięki temu goście i użytkownicy będą mieli łatwy dostęp do potrzebnych informacji, na podstawie których będą mogli sporządzać wykresy i wyciągać wnioski na temat podobieństw danych gatunków. W pierwszym ekranie użytkownik będzie mógł się zalogować, następnie wyświetli się ekran główny z tabelą z danymi i opcjami które można wybrać by na nich operować. Wykonane obliczenia będą wyświetlać się na ekranie. Dane te będzie można wyeksportować w formie raportu do pliku.

Niniejsza dokumentacja projektu inżynierskiego skupi się na przedstawieniu szczegółów dotyczących tworzenia oraz funkcjonowania systemu wraz z omówieniem technologii wykorzystanych do stworzenia jej. W kolejnych rozdziałach znajdą się informacje

na temat założeń projektowych, opisu funkcjonalności, użytych technologii, architektury, projektowania, tworzenia i testowania systemu.

2. Opis funkcjonalności

Oto lista funkcjonalności, które będą dostępne w aplikacji "Pynalise":

- Logowanie użytkowników - system będzie wymagał, aby użytkownicy się zalogowali przed uzyskaniem dostępu do danych.
- Role użytkowników - aplikacja będzie mieć trzy role: użytkownik, gość i administrator. Każda rola będzie miała dostęp do określonych funkcji.
- Przeglądanie danych - goście i użytkownicy będą mogli przeglądać dane, domyślne dane będą dotyczyły zwierząt w Zoo: gatunek, czy zwierzę poluje, czy składa jaja, czy posiada futro i wiele podobnych..
- Dodawanie danych - użytkownicy będą mogli dodawać nowe dane wczytując je z plików CSV lub JSON.
- Edytowanie danych - użytkownicy będą mogli edytować dane w tabeli.
- Usuwanie danych - użytkownicy będą mogli usuwać dane w tabeli.
- Zarządzanie użytkownikami - administratorzy będą mieli dostęp do funkcji zarządzania użytkownikami, takich jak: dodawanie nowych użytkowników, usuwanie użytkowników, edycja danych osobowych użytkowników.
- Raportowanie - aplikacja będzie miała możliwość generowania raportów dotyczących danych. Raporty będą dostępne w formie plików PDF.
- Statystyki - aplikacja będzie miała możliwość generowania statystyk dotyczących atrybutów danych oraz generowania wykresów dla porównania poszczególnych atrybutów.

Te funkcjonalności pozwolą użytkownikom na przeglądanie, zarządzanie i analizowanie danych w sposób prosty i intuicyjny.

3. Technologie:

Aplikacja będzie desktopowa. W projekcie wykorzystam następujące technologie: Do stworzenia GUI użytkownika wykorzystam technologię języka programowania Python. Bazę danych stworzę na serwerze MySQL. Żeby ukazać lepiej możliwości technologii Python stworzę również stronę internetową używając Framework Django. Użytkownicy będą mogli pobrać program, zgłaszać błędy dotyczące aplikacji oraz przeczytać dokumentację. Wybór technologii jest uzasadniony ze względu na ich popularność, wsparcie społeczności oraz możliwość zintegrowania aplikacji desktopowej z witryną internetową. To podejście może znacząco zwiększyć użyteczność aplikacji i ułatwić jej rozwijanie oraz utrzymanie.

3.1. Wybór typu aplikacji:

Wybór typu aplikacji jest ważną decyzją i zależy od wielu czynników. Podczas wyboru typu aplikacji kierowałem się następującymi argumentami:

- **Wydajność:** Aplikacje desktopowe często oferują wyższą wydajność niż aplikacje webowe lub mobilne, ponieważ działają na komputerze lokalnym, wykorzystując pełną moc obliczeniową urządzenia
- **Funkcjonalność:** Aplikacje desktopowe mają tendencję do oferowania bardziej zaawansowanych funkcji i możliwości, ponieważ nie są ograniczone przez przeglądarkę internetową lub ograniczenia systemu operacyjnego mobilnego
- **Dostęp do zasobów lokalnych:** Aplikacje desktopowe mogą mieć dostęp do zasobów lokalnych, takich jak pliki i dane, co może być ważne w niektórych przypadkach, takich jak praca z dużymi plikami
- **Bezpieczeństwo:** Aplikacje desktopowe mogą być bardziej bezpieczne niż aplikacje webowe, ponieważ dane i operacje są przechowywane na komputerze użytkownika, a nie w chmurze.
- **Brak potrzeby dostępu do internetu:** Aplikacje desktopowe nie zawsze wymagają stałego dostępu do internetu, co może być korzystne w sytuacjach, gdzie łączność internetowa jest niestabilna

Kierując się tymi punktami doszedłem do wniosku, że aplikacja do analizy danych, która może potrzebować dużej mocy obliczeniowej, pamięci, dostępu do plików z urządzenia będzie najlepiej funkcjonować jako aplikacja desktopowa. Jednak istnieją argumenty przemawiające za tym, że aplikacja webowa lub mobilna może być lepszym wyborem, na przykład:

- **Dostępność na wielu urządzeniach:** Aplikacje webowe i mobilne są bardziej dostępne na różnych urządzeniach, co jest przydatne, użytkownik może korzystać z aplikacji na smartfonie, tablecie i komputerze
- **Aktualizacje i utrzymanie:** Aplikacje webowe i mobilne są łatwiejsze do aktualizacji i utrzymania, ponieważ zmiany mogą być wprowadzane centralnie na serwerze, bez konieczności aktualizowania oprogramowania na każdym urządzeniu użytkownika
- **Udostępnianie:** Aplikacje webowe są łatwiejsze do udostępniania, ponieważ nie wymagają instalacji na każdym urządzeniu, a dostęp można uzyskać przez przeglądarkę internetową

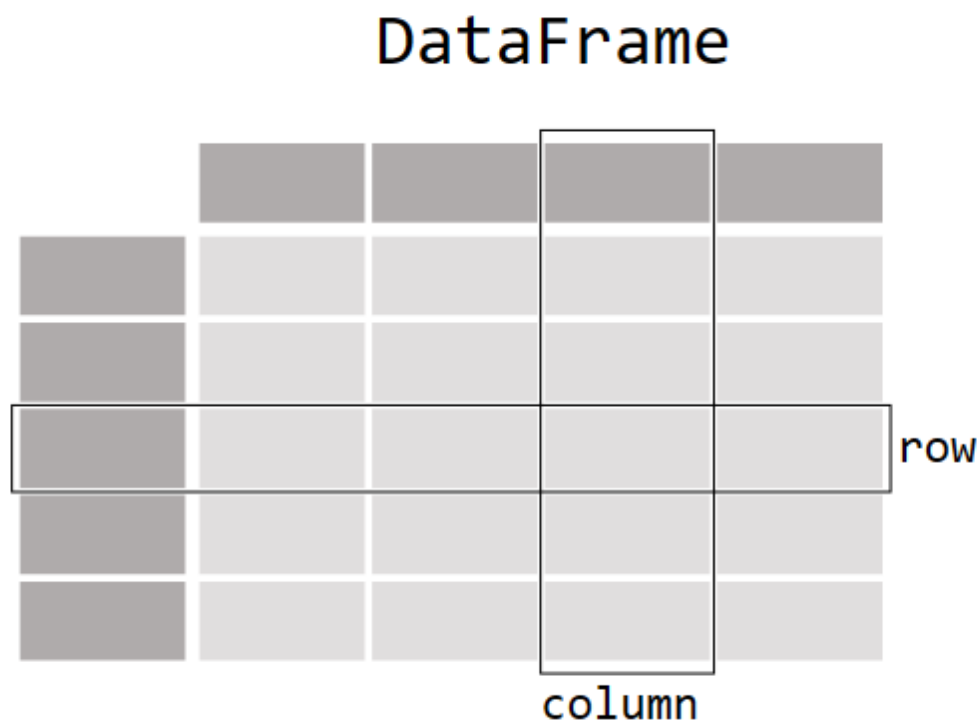
Z tego w związku żeby rozwiązać niektóre z tych problemów postanowiłem stworzyć również prostą stronę internetową, która będzie umożliwiać pobranie aktualnej wersji aplikacji z wielu urządzeń, możliwość zgłoszenia błędu oraz dostęp do dokumentacji projektu.

3.2. Biblioteki Pythona wykorzystane w projekcie aplikacji desktopowej:

3.2.1. Pandas

Jest to ogólnodostępna biblioteka, otwarte oprogramowanie wykorzystywane w analizie oraz przetwarzaniu danych. Posiada ona licencję BSD. Jest to elastyczna i permissywna licencja, która daje nieograniczone prawa do użytkowania, kopiowania, modyfikacji i redystrybucji kodu źródłowego. Najważniejsze funkcjonalności:

- **DataFrame** – jest to struktura danych i najważniejsze narzędzie oferowane przez to oprogramowanie. Jest to tabela przechowująca dane posiadająca wiersze oraz kolumny do których odpowiednie funkcje dają nam łatwy dostęp oraz możliwość modyfikacji.



Rysunek 1. DataFrame

- Wczytywanie oraz zapisywanie danych jest kompatybilne z różnymi formatami plików oraz źródłami danych (CSV, Excel, JSON, Parquet)
- Pozwala dzielić oraz filtrować dane używając odpowiednio napisanych warunków
- Umożliwia tworzenie nowych kolumn w oparciu o już istniejące
- Współdziała z biblioteką Matplotlib, dzięki czemu można generować wykresy w oparciu o dane zawarte w tabeli
- Zawiera w sobie funkcje umożliwiające obliczenia statystyczne

- Umożliwia pracę z różnymi rodzajami danych, chociażby wspiera pracę z danymi czasowymi, takimi jak daty czy godziny
- Pozwala zachować kontrolę nad czystością danych, poprzez sprawdzanie czy komórki są puste lub wypełnienie ich odpowiednimi wartościami

3.2.2. Matplotlib

To wszechstronna wielofunkcyjna biblioteka służąca do generowania statycznych, ruchomych, interaktywnych wykresów oraz wizualizacji danych. Szeroko wykorzystuje się ją w analizie danych, analizie finansowej, badaniach naukowych czy też w inżynierii.

Najważniejsze funkcjonalności Matplotlib:

- Konfigurowalność, czyli możliwość dostosowania wyglądu. Można kontrolować kolorystykę, typy linii, etykiety osi, tytuły, legendy i wiele innych aspektów wykresu
- Wsparcie dla wielu formatów wyjściowych: Umożliwia zapisywanie wykresów w różnych formatach, takich jak PNG, JPEG, SVG, PDF i wiele innych. Dzięki temu można łatwo udostępniać swoje wizualizacje w różnych kontekstach
- Integracja z innymi narzędziami: Matplotlib może być używane w połączeniu z innymi bibliotekami Python, takimi jak NumPy do przetwarzania danych numerycznych, pandas do analizy danych i SciPy do obliczeń naukowych
- Tworzenie różnych rodzajów wykresów:
 - Wykres liniowy (Line Plot): Wykres przedstawiający dane za pomocą linii, które łączą punkty danych
 - Wykres punktowy (Scatter Plot): Wykres przedstawiający punkty danych na płaszczyźnie, gdzie każdy punkt reprezentuje parę wartości (x, y)
 - Wykres słupkowy (Bar Plot): Wykres przedstawiający dane za pomocą słupków o różnej długości, które reprezentują wartości w różnych kategoriach
 - Wykres kołowy (Pie Chart): Wykres przedstawiający dane w formie koła, gdzie każdy kawałek koła reprezentuje procentową część całkowitej wartości
 - Histogram: Wykres przedstawiający rozkład danych numerycznych na podstawie częstotliwości występowania w określonych przedziałach
 - Wykres konturowy (Contour Plot): Wykres przedstawiający dane przestrzenne za pomocą kontur, które reprezentują równość wartości

- Wykres heatmap (Mapa cieplna): Wykres przedstawiający dane w postaci kolorowych kwadratów na siatce, gdzie kolor reprezentuje wartość w danym punkcie
- Wykres pudełkowy (Box Plot): Wykres przedstawiający rozkład danych w formie pudełka z wąsami, co pomaga w analizie rozkładu danych i identyfikacji wartości odstających
- Wykres skrzypcowy (Violin Plot): Wykres łączący wykres pudełkowy z histogramem, umożliwiający lepsze zrozumienie rozkładu danych
- Wykres słupkowy skumulowany (Stacked Bar Plot): Wykres przedstawiający dane za pomocą słupków, w których różne wartości są nakładane na siebie
- Wykres powierzchniowy (3D Surface Plot): Wykres przedstawiający dane przestrzenne w trójwymiarowej przestrzeni za pomocą powierzchni
- Wykres konturowy trójwymiarowy (3D Contour Plot): Wykres przedstawiający dane przestrzenne w trójwymiarowej przestrzeni za pomocą kontur

3.2.3. PyQT6

To złożona biblioteka do tworzenia interfejsu użytkownika dla aplikacji desktopowych opartych na widgetach umożliwiająca obsługę zdarzeń. QT6 to szósta wersja Frameworka stworzonego w języku C++ do tworzenia aplikacji, obsługi ich zdarzeń, komunikacji sieciowej czy wymiany plików. Został on powiązany z językiem Python i w ten sposób powstała biblioteka PyQT6, która umożliwia wykorzystanie kodu źródłowego napisanego domyślnie w C++ w programach tworzonych w technologii Python. PyQT6 posiada licencję GPL v3 (General Public License), która umożliwia swobodne modyfikowanie i rozpowszechnianie tego oprogramowania. Najważniejsze zalety tego narzędzia:

- Obsługa zdarzeń jest jedną z najważniejszych funkcjonalności, dzięki niej możemy kontrolować aplikacje za pomocą kliknięć myszy, naciskając poszczególne przyciski, komponenty, możemy również wprowadzać dane z klawiatury.
- Wieloplatformowość: Aplikacje stworzone w Qt mogą działać na różnych systemach operacyjnych, takich jak Windows, macOS i Linux.
- Qt Designer – jest to narzędzie do tworzenia interfejsu, które bardzo ułatwia projektowanie aplikacji. Można tam układać komponenty metodą przeciągania oraz upuszczania.
- Qt obsługuje wiele języków programowania. Przede wszystkim C++ w którym został napisany oraz Python przy pomocy PyQT6, jednak są również wiązki umożliwiające pracę w innych językach jak Java, Ruby, C#, Perl czy nawet JavaScript
- Kompleksowa dokumentacja i wiele Źródeł informacji ze względu na dużą społeczność posługującą się tym narzędziem

3.2.4. scikit-learn

Biblioteka służąca to wykorzystywaniu algorytmów uczenia maszynowego. To kompleksowe narzędzie wykorzystujące NumPy, matplotlib czy SciPy. Posiada licencję BSD podobnie jak pandas. Stworzona została w 2007 roku przez Davida Cournapeau jako Google Summer of Code project. Wielu ludzi o tamtego czasu rozwinęło ten projekt dokładając swoje spostrzeżenia oraz doświadczenie. Najważniejsze cechy tego oprogramowania:

- **Wszechstronność:** Scikit-learn zawiera bogatą kolekcję algorytmów uczenia maszynowego, takich jak algorytmy klasyfikacji, regresji, grupowania, redukcji wymiarowości, wykrywania anomalii, nawiasowe, klasteryzacji, uczenie semi-nadzorowane, uczenie wzmacniane, analiza skupień
- **Łatwość użycia:** Biblioteka jest zaprojektowana w sposób intuicyjny i łatwy do nauki. Oferuje spójne API dla różnych algorytmów, co ułatwia korzystanie z różnych modeli
- **Wsparcie dla danych:** Scikit-learn obsługuje dane w postaci tabelarycznych struktur danych, takich jak tablice NumPy i ramki danych pandas, co ułatwia pracę z danymi
- **Ocena modeli:** Biblioteka dostarcza narzędzia do oceny i walidacji modeli, takie jak podział danych na zbiory treningowe i testowe, walidacja krzyżowa, oraz różne metryki oceny wydajności modeli
- **Preprocessing danych:** Scikit-learn oferuje narzędzia do przetwarzania danych, w tym standaryzację, normalizację, kodowanie zmiennych kategorycznych i wiele innych
- **Wizualizacje:** Biblioteka umożliwia wizualizację wyników analizy i modeli za pomocą wykresów
- **Dokumentacja i społeczność:** Scikit-learn posiada obszerną dokumentację, która zawiera przykłady użycia i porady. Ponadto, jest wspierane przez aktywną społeczność użytkowników i programistów

W moim projekcie użyję wybrane algorytmy oraz omówię ich modele matematyczne.

3.2.5. reportlab

To biblioteka, która używana jest do generowania dokumentów oraz grafik w Pythonie. Posiada licencję BSD podobnie jak pandas oraz scikit-learn. Stworzył ją Andy Robinson a następnie rozwijana była przez społeczność. W tym projekcie zostanie wykorzystana do generowania raportów PDF z wynikami obliczeń.

3.2.6. Seaborn

Rozwinięcie biblioteki matplotlib, pozwala na generowanie atrakcyjnych dla oka wykresów oraz grafik statystycznych. Zostanie użyta by lepiej oddać niektóre wykresy. Stworzona została przez Michaela Waskoma w 2013 roku i jest jednym z najpopularniejszych narzędzi wykorzystywanych w tej dziedzinie.

3.2.7. JSON

Jest to standardowy moduł Pythona pozwalający na kodowanie oraz dekodowanie danych w formacie .JSON.

3.3. Biblioteki Pythona wykorzystane w stronie internetowej:

3.3.1. Django

3.3.2. bootstrap4 for django

3.4. Środowisko programistyczne

Do stworzenia mojego projektu wybrałem środowisko Pycharm. Jest to bardzo popularne oraz rozbudowane IDE (Integrated Development Enviroment). Oferuje ono wiele funkcjonalności, które sprawiają że praca w nim jest niezwykle skuteczna:

- Inteligentne podpowiedzi – oprogramowanie zapobiega pisania błędów analizując nasz kod na podstawie zainstalowanych domyślnie oraz zewnętrznie bibliotek
- zaawansowane narzędzia do debugowania pozwalają diagnozować skomplikowane problemy w kodzie i pomagają je rozwiązywać
- jest to narzędzie zintegrowane z Gitem, co pozwala łatwo zabezpieczać kod oraz pracować na nim w grupie
- Wspiera różne frameworki takie jak Django, Flask, Pyramid i wiele innych. Dzięki temu można tworzyć aplikacje webowe i inne projekty oparte na frameworkach.
- Pozwala tworzyć i porządkować duże projekty integrując przy tym różne technologie, obsługuje również inne języki, nie tylko Python
- Rozbudowane mechanizmy wyszukiwania, które pozwalają skutecznie odnaleźć się nawet w bardzo złożonym projekcie

3.5. System kontroli wersji

Oprogramowaniem, które posłuży mi do zapisywania stanu mojego projektu będzie GIT. Jest szeroko stosowany w dziedzinie zarządzania wersjami i kontroli kodu źródłowego w projektach informatycznych. Główną funkcją Gita jest umożliwienie zespołowi programistów pracę na jednym projekcie. Użytkownik tworzy własne repozytorium z kodem do którego umożliwia dostęp i możliwość edycji innym użytkownikom za pomocą poleceń typu PUSH, COMMIT lub PULL. Deweloper może również opublikować kod swojego programu publicznie dzięki czemu różni ludzie mogą go pobrać, rozwinąć, znaleźć błędy lub po prostu używać. Jest to bardzo dobre narzędzie do śledzenia zmian w projekcie czy tworzenia kopii zapasowych, ponieważ można cofnąć stan projektu do każdej jego aktualizacji. Z tego względu postanowiłem go wykorzystać do pracy z moim projektem, żeby zabezpieczyć swój program.

4. Modele Matematyczne zaimplementowane w aplikacji

Matematyka jest silnie związana z programowaniem. Zanim przełożymy nasz problem na kod warto jest przeanalizować go pod względem matematycznym dzięki czemu nasze rozwiązanie będzie przemyślane i skuteczne. W kolejnych podpunktach przedstawię modele matematyczne, które zostaną zaimplementowane w moim systemie. Przedstawię obliczenia ich złożoności obliczeniowej oraz pamięciowej.

4.1. Klasyfikacja – model algorytmu regresji liniowej

4.2. Klasyfikacja – model algorytmu drzewa decyzyjnego

4.3. Klasteryzacja – model algorytmu k-sąsiadów

4.4. Minimum

Do obliczenia tej wartości wykorzystujemy wzór na minimalną liczbę z dwóch wybranych zmiennych.

$$\min = \frac{1}{2}(x_1 + x_2 - |x_1 - x_2|)$$

Wzór jest prosty w działaniu. Złożoność obliczeniowa takiego równania to $O(1)$. Jednak do obliczenia minimalnej wartości dla całej kolumny danych wykorzystamy funkcję `min()` która wyznacza minimalną wartość z całej tablicy poprzez porównywanie po kolei wartości i zapisanie najmniejszej. Wtedy złożoność obliczeniowa wzrasta do $O(n)$ ze względu na ilość danych wejściowych, która jest elastyczna.

4.5. Maximum

Analogicznie dla wartości minimalnej obliczamy wartość maksymalną.

$$\max = \frac{1}{2}(x_1 + x_2 + |x_1 - x_2|)$$

Do obliczenia jej w systemie dla całego zbioru danych wykorzystamy funkcję `max()` z wbudowanej biblioteki Pythona.

4.6. Odchylenie Standardowe

Oznaczmy sobie tę statystykę jako zmienną σ . Do wyznaczenia tej wartości wykorzystamy również średnią arytmetyczną \bar{x} . n to suma elementów w zbiorze. Odchylenie standardowe tych liczb od ich średniej arytmetycznej, to pierwiastek kwadratowy z wariancji.

$$\sigma = \sqrt{\frac{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \dots + (x_n - \bar{x})^2}{n}}$$

Do policzenia tego równania wykorzystamy funkcję `std()` z wbudowanej biblioteki statystycznej Pythona.

4.7. Średnia arytmetyczna

Średnią oznaczmy sobie przez \bar{x} . wyliczamy ją obliczając sumę wszystkich elementów od x_1 do x_n , a następnie tę sumę należy podzielić przez liczbę elementów w zbiorze czyli n . Oto wzór matematyczny reprezentujący ten model:

$$\bar{x} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{n}$$

Złożoność obliczeniowa tego modelu wynosi $O(n)$, jest to model liniowy. Do obliczania tej funkcji w systemie wykorzystamy wbudowaną funkcję w pythonie `mean()`, która korzysta z tego wzoru.

4.8. Korelacja

Stworzymy sobie model matematyczny korelacji dla dwóch zmiennych x oraz y . Wykorzystamy do tego wzór na współczynnik korelacji Pearsona. Ten współczynnik mówi nam jaka jest siła i kierunek zależności liniowej pomiędzy naszymi zmiennymi.

Współczynnik r przyjmuje wartości z przedziału $[-1,1]$,

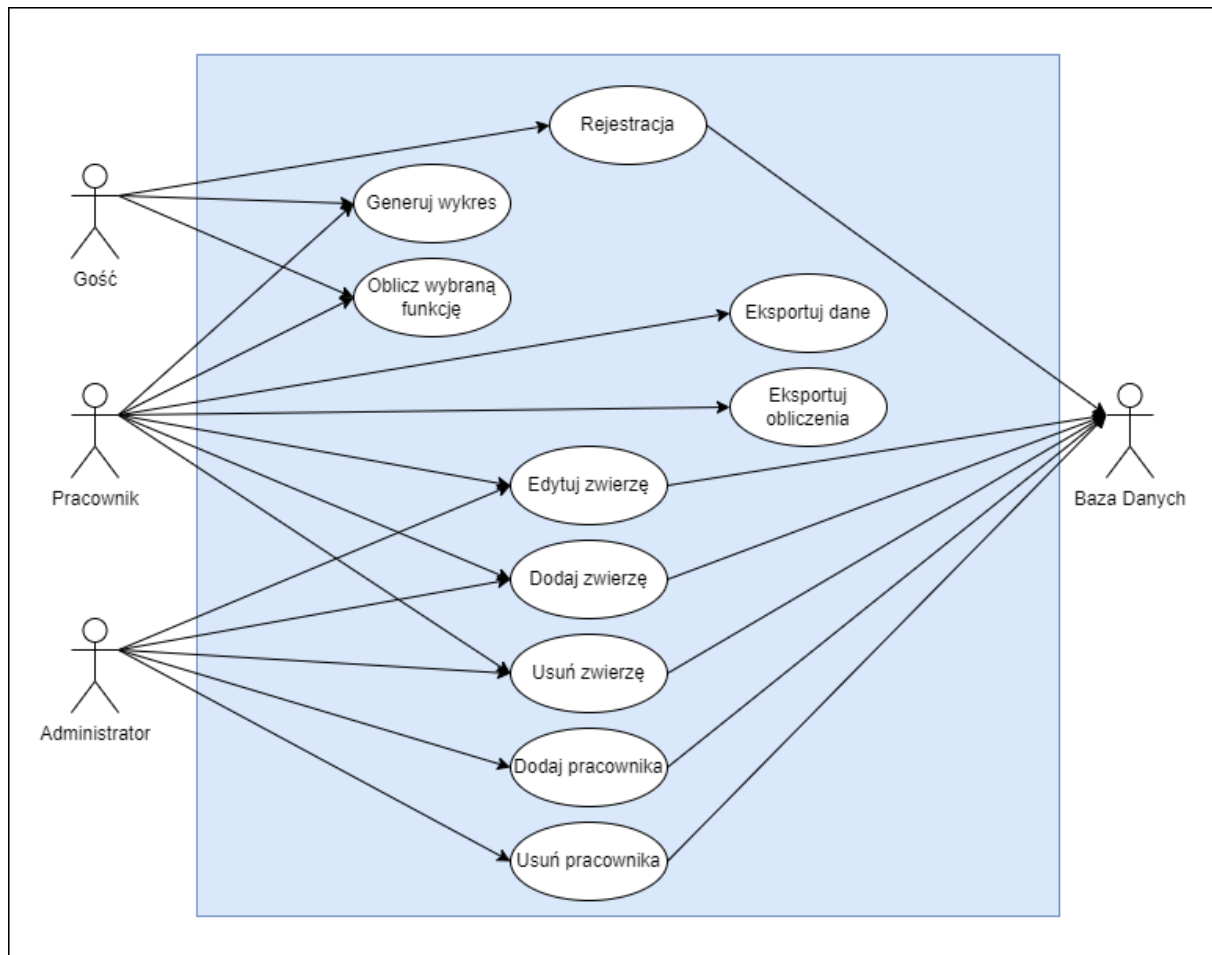
- Im wartość bliższa 1 tym zależność jest silniejsza i dodatnia (jeżeli x rośnie to y rośnie),
- Im wartość bliższa -1 tym zależność jest silniejsza i ujemna (jeżeli x rośnie to y maleje),
- $r=0$ oznacza brak związku liniowego pomiędzy zmiennymi.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$

Do obliczania korelacji w naszym systemie wykorzystamy funkcję `corr()`, która oblicza macierz współczynników korelacji Pearsona dla wybranych kolumn w tabeli.

5. Diagramy UML

5.1. Diagram przypadków użycia:



• Opis Aktorów

Gość - osoba mogąca jedynie wyświetlać informacje w tabeli, może generować wykresy i obliczać funkcje statystyczne,

Użytkownik - może używać funkcji gościa, ale dodatkowo może eksportować dane oraz obliczenia do pliku zewnętrznego. Użytkownik może również modyfikować listę zwierząt.

Administrator – ma dostęp do całej bazy danych zwierząt oraz może zarządzać użytkownikami.

Baza Danych – przechowuje wszystkie informacje na temat użytkowników oraz zwierząt.

- Dokumentacja przypadków użycia:

Nazwa:	Rejestracja
Identyfikator:	P01
Aktorzy:	Gość, Baza Danych
Krótki opis:	Rejestracja użytkownika
Warunki wstępne:	Gość nie ma wcześniej założonego konta w systemie.
Warunki końcowe:	Gość staje się użytkownikiem w systemie po pomyślnym ukończeniu rejestracji.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Gość wybiera odpowiednią opcję, aby rozpocząć rejestrację. 2. Gość uzupełnia wszystkie pola, które są obowiązkowe aby ukończyć rejestrację. 3. System dodaje gościa jako nowego użytkownika. 4. Gość może zalogować się na stronie na swoje nowe konto jako użytkownik.
Alternatywne przepływy zdarzeń:	2a. Gość podał nieprawidłowe dane. 3a. Brak połączenia z bazą danych i system nie może dodać nowego użytkownika.
Specjalne wymagania:	Połączenie z bazą danych systemu

Nazwa:	Generuj Wykres
Identyfikator:	P02
Aktorzy:	Gość, Użytkownik
Krótki opis:	Generuje wykres
Warunki wstępne:	Wybrane jeden lub dwa atrybuty
Warunki końcowe:	Otwiera się nowe okno z wygenerowanym wykresem
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik wybiera jeden lub dwa atrybuty w oparciu o które aplikacja ma wygenerować wykres. 2. Użytkownik naciska przycisk który generuje wykres. 3. Otwiera się nowe okno wyświetlające wykres. 4. Użytkownik może eksportować, zapisać, analizować lub zamknąć wykres.
Alternatywne przepływy zdarzeń:	2a. Atrybuty wybrane przez użytkownika są takie same co skutkuje wygenerowaniem wykresu w oparciu tylko o jeden atrybut. 3a. Jeśli dane nie będą spełniać wymagań zaimplementowanego wykresu nie zostanie on wygenerowany.

Specjalne wymagania:	Specjalna biblioteka do generowania wykresów
-----------------------------	--

Nazwa:	Oblicz wybraną funkcję
Identyfikator:	P03
Aktorzy:	Gość, Użytkownik
Krótki opis:	Oblicza funkcję statystyczną dla wybranego atrybutu
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Wyświetla się obliczony wynik w oknie do wyświetlania obliczeń
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik wybiera dla którego atrybutu ma zostać obliczona funkcja poprzez zaznaczenie kolumny w tabeli. 2. System oblicza daną funkcję. 3. Wynik zostaje wyświetlony w oknie do obliczeń.
Alternatywne przepływy zdarzeń:	2a. Jeśli dane nie będą liczbą tylko ciągiem znaków zostanie wypisany błąd
Specjalne wymagania:	Specjalna biblioteka do obliczania różnych funkcji statystycznych

Nazwa:	Eksportuj Dane
Identyfikator:	P04
Aktorzy:	Użytkownik
Krótki opis:	Eksportuje dane do pliku PDF
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Wyświetla się komunikat o tym że plik został poprawnie wyeksportowany oraz zapisany.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik decyduje się na eksportację danych i naciska przycisk. 2. Użytkownik wybiera gdzie plik ma zostać zapisany na dysku. 3. Użytkownik zostaje poinformowany o prawidłowym wyeksportowaniu danych i zapisaniu pliku.
Alternatywne przepływy zdarzeń:	2a. Jeśli użytkownik nie ma miejsca na dysku plik nie zostanie zapisany.
Specjalne wymagania:	Komponent do wyboru miejsca na dysku.

Nazwa:	Eksportuj Obliczenia
Identyfikator:	P05
Aktorzy:	Użytkownik
Krótki opis:	Eksportuje obliczenia do pliku PDF
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Wyświetla się komunikat o tym że plik został poprawnie wyeksportowany oraz zapisany.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik decyduje się na eksportację wyników i naciska przycisk. 2. Użytkownik wybiera gdzie plik ma zostać zapisany na dysku. 3. Użytkownik zostaje poinformowany o prawidłowym wyeksportowaniu danych i zapisaniu pliku.
Alternatywne przepływy zdarzeń:	2a. Jeśli użytkownik nie ma miejsca na dysku plik nie zostanie zapisany.
Specjalne wymagania:	Komponent do wyboru miejsca na dysku.

Nazwa:	Edytuj dane
Identyfikator:	P06
Aktorzy:	Użytkownik, Administrator
Krótki opis:	Edytuje dane w tabeli
Warunki wstępne:	Naciśnięty odpowiedniej komórki w tabeli
Warunki końcowe:	Dane wyświetlane w tabeli się zmieniają
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik naciska komórkę 2. Użytkownik wypełnia komórkę nową informacją. 3. Użytkownik zatwierdza zmiany. 4. Zmiany przesłane i zapisane są w bazie danych.
Alternatywne przepływy zdarzeń:	<p>3a Jeśli pola zostały wypełnione nieprawidłowymi danymi zostanie wypisany odpowiedni komunikat.</p> <p>4a. Jeśli nie ma połączenia z serwerem bazy danych wystąpi błąd.</p>
Specjalne wymagania:	Połączenie z bazą danych

Nazwa:	Wczytaj Dane
Identyfikator:	P07
Aktorzy:	Użytkownik, Administrator
Krótki opis:	Wczytuje dane z pliku CSV lub JSON do tabeli
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Dane zostają dopisane do tabeli.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik naciska przycisk wczytania danych 2. Użytkownik wypełnia odpowiednie pola. 3. Użytkownik zatwierdza decyzje. 4. Dane przesłane i zapisane są w bazie danych.
Alternatywne przepływy zdarzeń:	<p>3a Jeśli pola zostały wypełnione nieprawidłowymi danymi zostanie wypisany odpowiedni komunikat.</p> <p>4a. Jeśli nie ma połączenia z serwerem bazy danych wystąpi błąd.</p>
Specjalne wymagania:	Połączenie z bazą danych

Nazwa:	Usuń dane
Identyfikator:	P08
Aktorzy:	Użytkownik, Administrator
Krótki opis:	Usuwa dane w tabeli i w bazie danych
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Odpowiedni wiersz w tabeli znika.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Użytkownik wybiera wiersz w tabeli z danym zwierzęciem. 2. Użytkownik naciska przycisk usuwania zwierzęcia 3. Odpowiedni wiersz znika z tabeli. 4. Analogiczne zmiany zachodzą w bazie danych.
Alternatywne przepływy zdarzeń:	4a. Jeśli nie ma połączenia z serwerem bazy danych wystąpi błąd.
Specjalne wymagania:	Połączenie z bazą danych

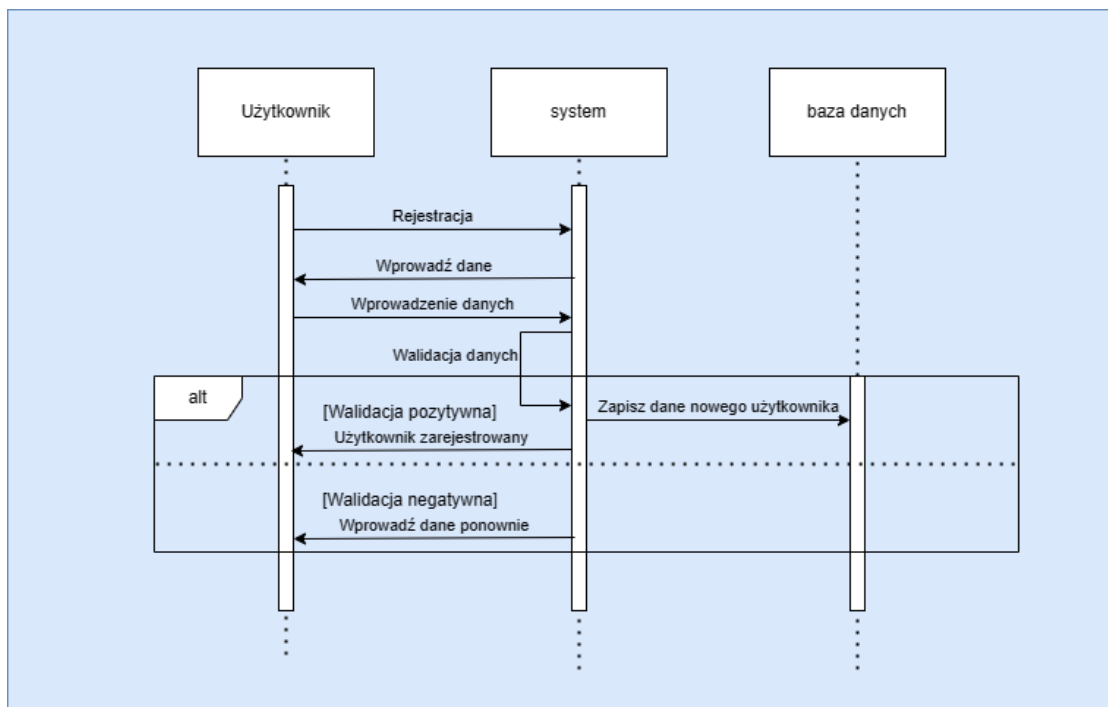
Nazwa:	Dodaj Użytkownika
Identyfikator:	P09
Aktorzy:	Administrator
Krótki opis:	Dodaje Użytkownika do bazy danych.

Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Nowy użytkownik zostaje dodany w tabeli wyświetlającej ich.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Administrator naciska przycisk dodania użytkownika. 2. Wypełnia odpowiednio pola. 3. Zatwierdza dane. 4. Dane zostają dodane do tabeli oraz do bazy danych.
Alternatywne przepływy zdarzeń:	<p>3a Jeśli pola zostały wypełnione nieprawidłowymi danymi zostanie wypisany odpowiedni komunikat.</p> <p>4a. Jeśli nie ma połączenia z serwerem bazy danych wystąpi błąd.</p>
Specjalne wymagania:	Połączenie z bazą danych

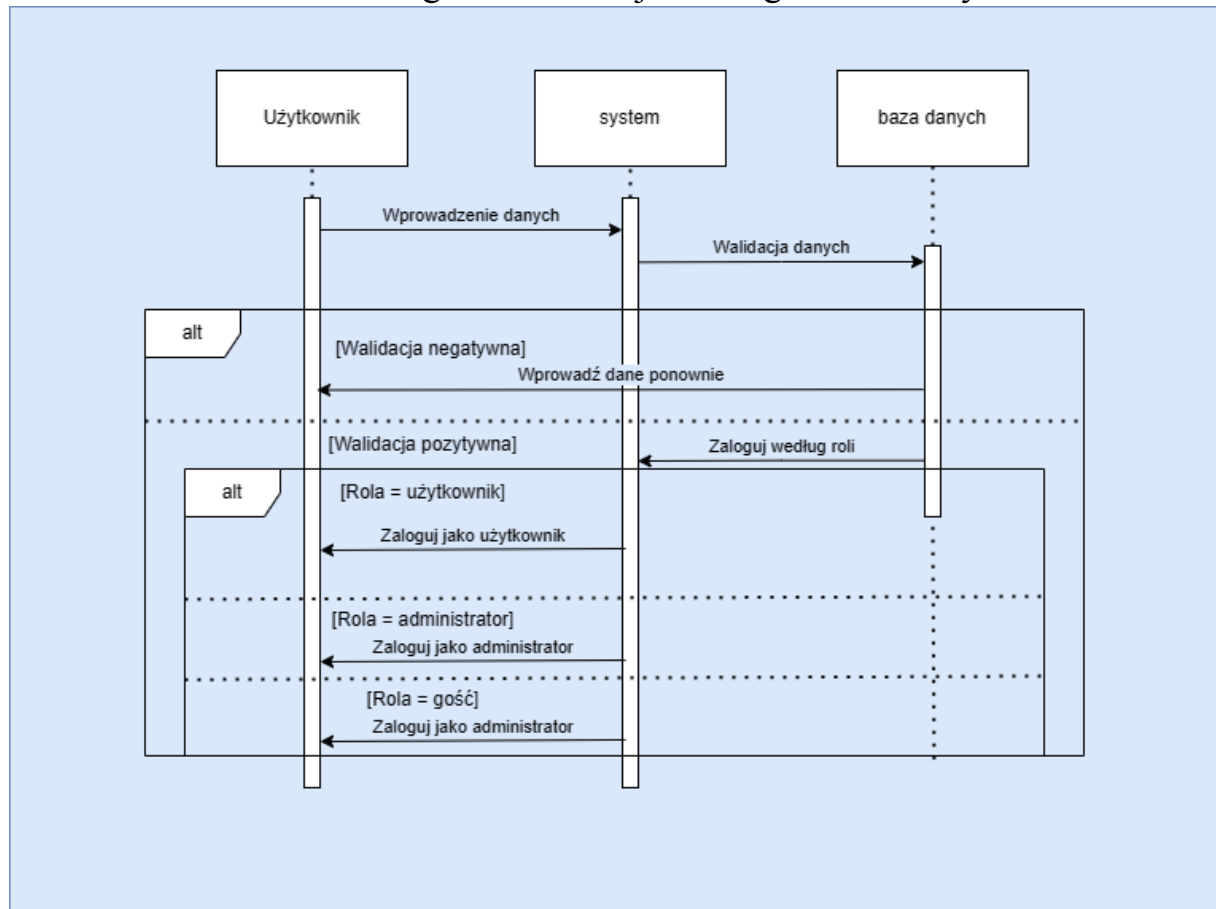
Nazwa:	Usuń Użytkownika
Identyfikator:	P10
Aktorzy:	Administrator
Krótki opis:	Usuwa użytkownika w tabeli i w bazie danych
Warunki wstępne:	Naciśnięty odpowiedni przycisk
Warunki końcowe:	Odpowiedni wiersz w tabeli znika.
Główny przepływ zdarzeń:	<ol style="list-style-type: none"> 1. Administrator wybiera wiersz w tabeli z danym Użytkownikiem. 2. Administrator naciska przycisk usuwania zwierzęcia 3. Odpowiedni wiersz znika z tabeli. 4. Analogiczne zmiany zachodzą w bazie danych.
Alternatywne przepływy zdarzeń:	4a. Jeśli nie ma połączenia z serwerem bazy danych wystąpi błąd.
Specjalne wymagania:	Połączenie z bazą danych

5.2. Diagramy sekwencji:

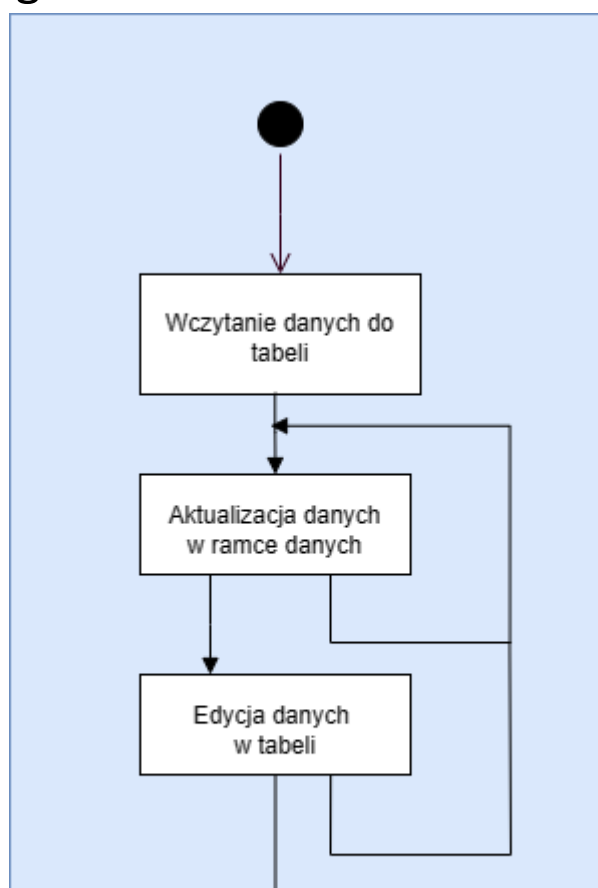
5.2.1. Diagram sekwencji dla Rejestracji nowego użytkownika



5.2.2. Diagram sekwencji dla Logowania do systemu



5.3. Diagram Stanów Tabeli



6. Proces tworzenia aplikacji

6.1. Projektowanie User Interface

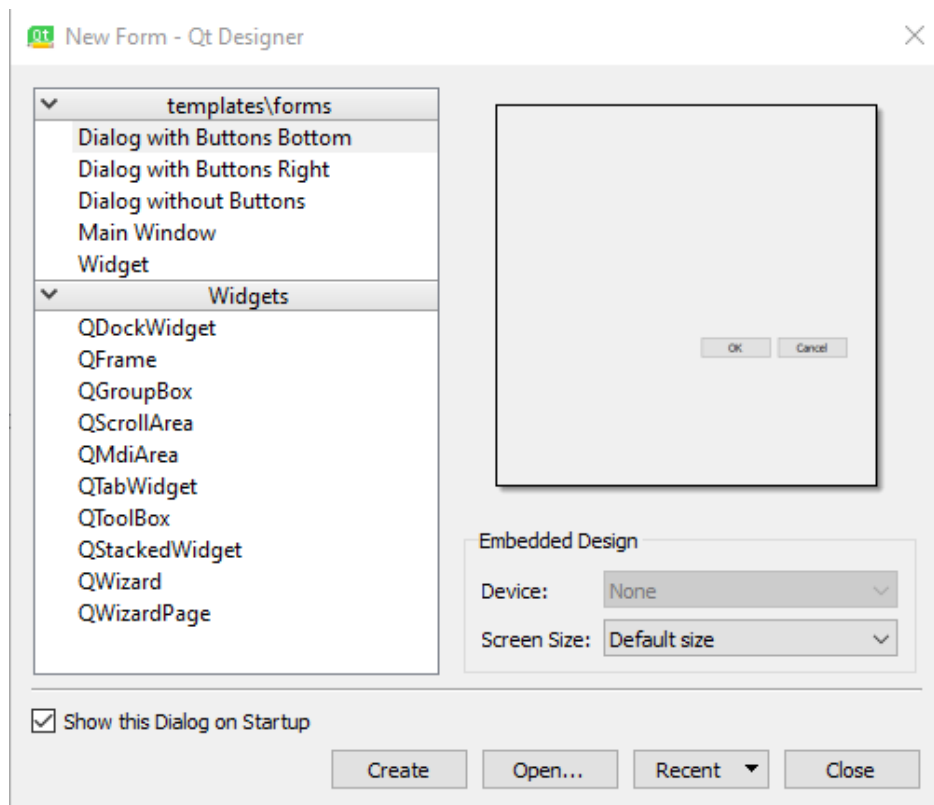
6.1.1. Przegląd dostępnych technologii na rynku.

Zacząłem szukać informacji o tym w jaki sposób najlepiej stworzyć interfejs użytkownika. Przejrzałem dostępne na rynku technologie. Pierwszą biblioteką do tworzenia UI na którą warto zwrócić uwagę jest Tkinter. Jest to standardowa technologia do tworzenia aplikacji w pythonie. Jej atutami jest prostota oraz wieloplatformowość, ponieważ działa ona różnych systemach operacyjnych. Kolejną biblioteką wartą uwagi jest Kivy. Jest to framework, który oprócz tego że pozwala tworzyć UI dla użytkowników windowsa czy Linusa pozwala również na programowanie interfejsów aplikacji mobilnych, animacji, elementów dotykowych czyli programów na system Adroid czy IOS.



6.1.2. Wybór PyQT6

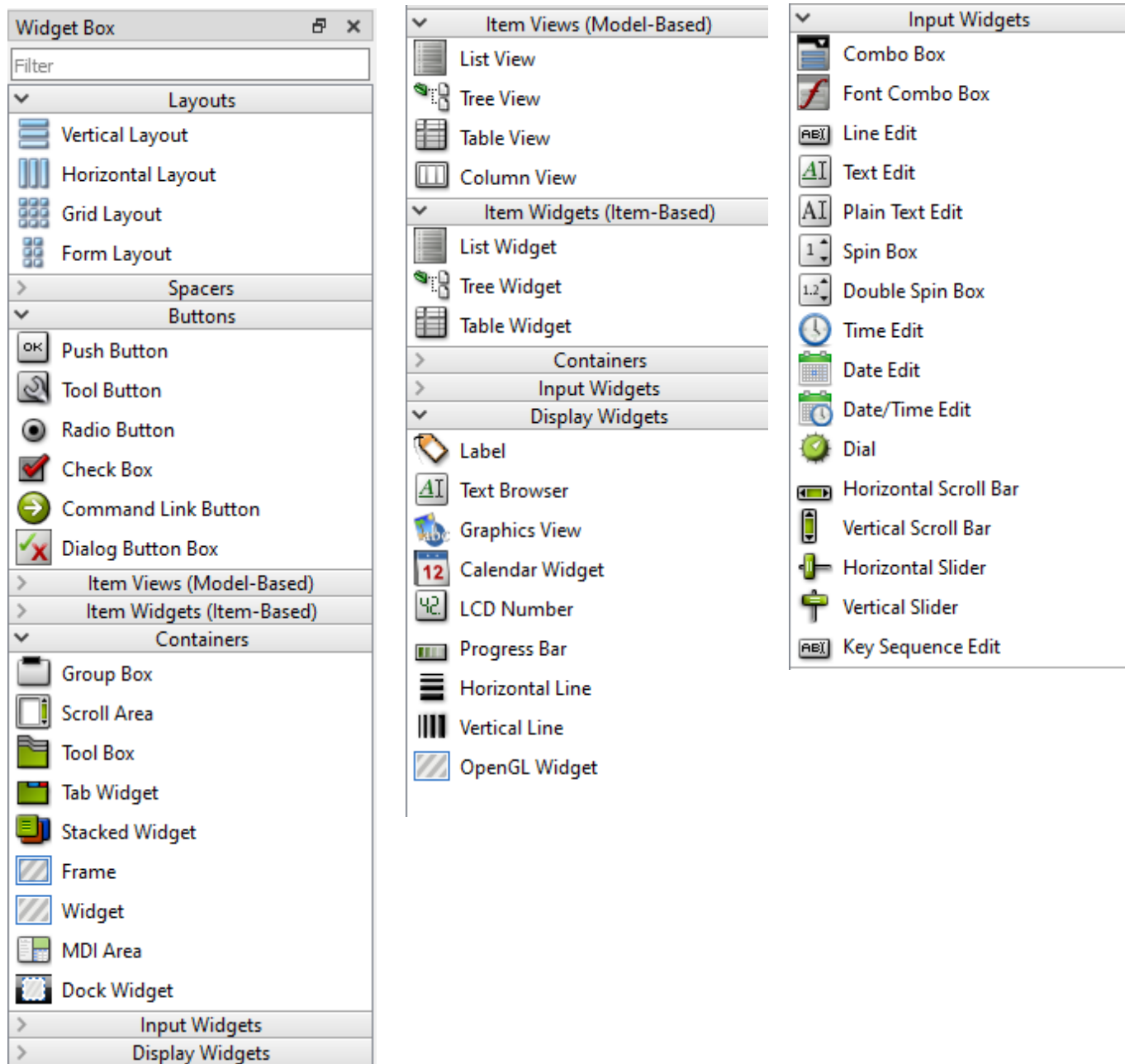
Ja jednak postanowiłem nie skorzystać z żadnej wyżej opisanych możliwości. Technologia którą wybrałem to PyQT6, która jest lepiej skonstruowana do tworzenia złożonych projektów niż inne technologie. Dodatkowo ma estetyczny wygląd. Funkcjonuje lepiej w kontekście wielu platform niż Tkinter. Lepiej też przystosowana jest do tworzenia aplikacji desktopowych niż Kivy, która skoncentrowana jest na aplikacjach mobilnych. Kolejnym aspektem który przemawia za używaniem PyQT6 jest QT Designer. Jest to narzędzie służące do projektowania aplikacji za pomocą przeciągania różnych komponentów w oknie projektowania aplikacji. Qt Designer jest niezależny od platformy i języka programowania. Nie tworzy kodu w żadnym konkretnym języku, ale tworzy pliki .ui. Pliki te mają rozszerzenie .XML ze szczegółowymi opisami generowania GUI opartych na Qt. Ja preferuję konwersję plików .ui na kod Pythona, ze względu na ujednolicenie kodu aplikacji co zmniejsza szansę na wystąpienie nieprzewidzianych błędów podczas eksportowania aplikacji użytkowej.



Omówimy teraz architekturę dostępnych w tej aplikacji okien. Dostępne do wyboru są trzy szablony okien: okno główne, okno dialogowe oraz widżet. Okno główne służy do wyświetlania najważniejszego widoku aplikacji. Jest to ten komponent, który posiada główne funkcjonalności całego programu, domyślnie posiada już pasek zadań, który można wykorzystać do własnych funkcji. Okna dialogowe są dodatkowe. Służą do podejmowania decyzji np. Jeśli chcemy zapytać użytkownika o dodatkowe informacje, możemy zaprojektować okno dialogowe w którym będzie je wprowadzał. Natomiast widżety są tak naprawdę wszystkim w QT. Wszystkie okna dialogowe, okna główne, komponenty, przyciski są tak naprawdę rozbudowanymi widżetami. Szablon widżetu pozwala nam zaprojektować własny, od podstaw, decydując o każdym jego aspekcie. Do wyboru mamy również szereg gotowych widżetów proponowanych przez oprogramowanie.

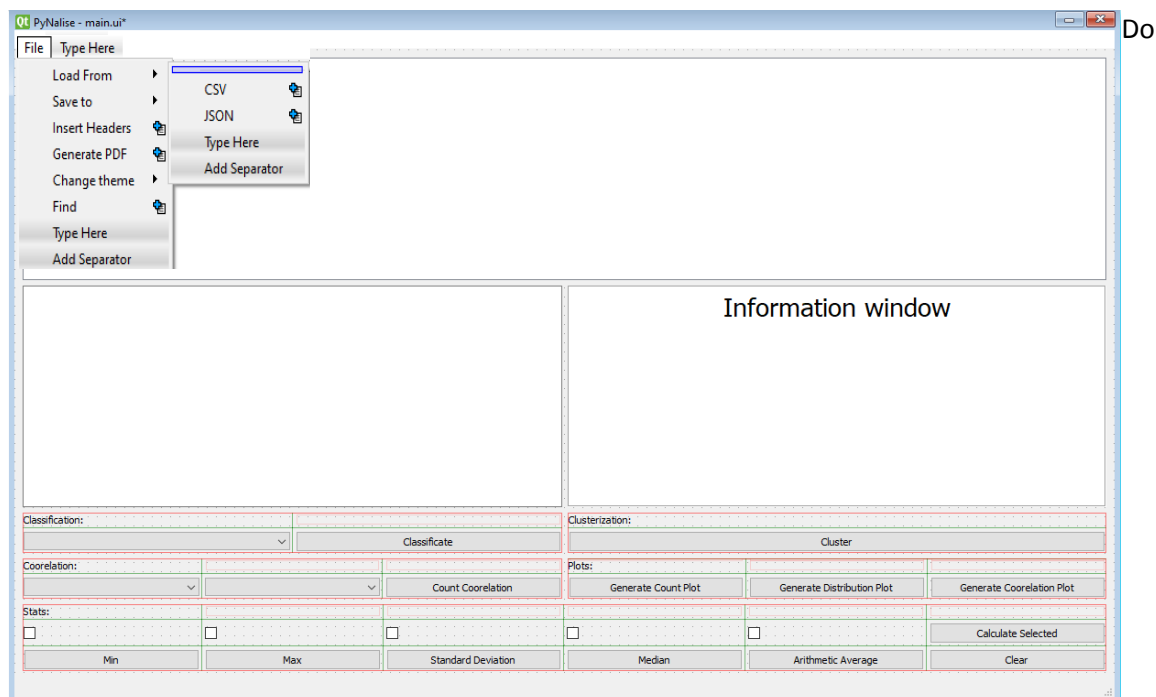
6.1.3. Projektowanie okien do Pyanalise.

Przedstawię teraz mój interfejs oraz proces tworzenia go krok po kroku używając zasobów opisanych powyżej. Pierwszym krokiem było zaplanowanie ilości okien w aplikacji. Przewidziałem trzy okna. Główne okno w którym wyświetlona zostanie tabela, przyciski oraz obliczenia, widżet logowania oraz widżet rejestracji. Kolejnym aspektem QT na który należało zwrócić uwagę jest „Widget Box”. W tej części oprogramowania do wykorzystania mamy gotowe komponenty, które posłużą nam do obsługi zdarzeń w naszej aplikacji, między innymi: kontenery, rozmieszczenia, przyciski, widoki do wyświetlania oraz wprowadzania danych.

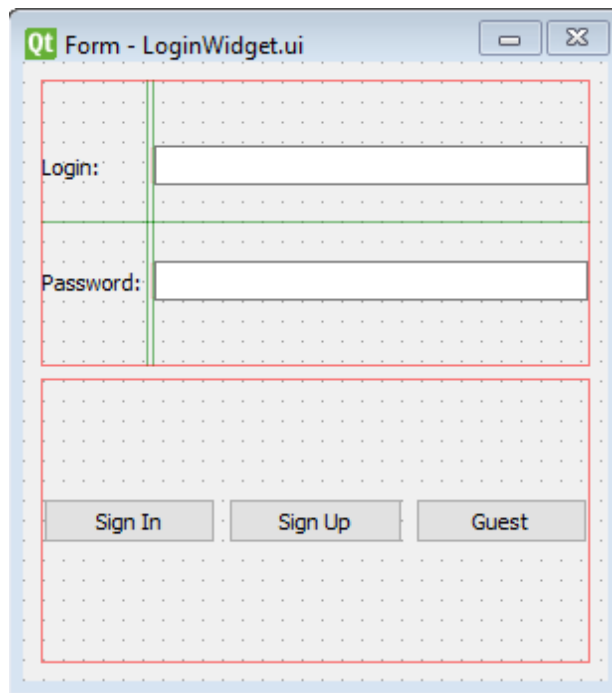


Pierwszym widżetem który użyję do projektu głównego okna jest rozmieszczenie GridLayout. Pozwala ono by program domyślnie ustawiał elementy w równej odległości, co ułatwia porządkowanie i poprawia estetykę programu. Widżety które użyję do stworzenia głównego okna mojego programu to:

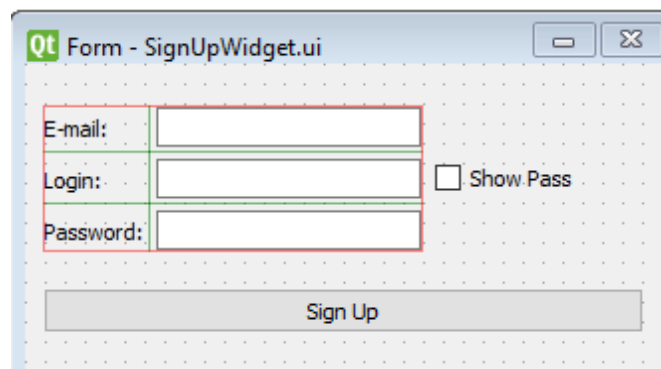
- TableView czyli widok tabeli z danymi.
- TextBrowser użyję do wyświetlania obliczeń oraz informacji na temat funkcji programu.
- Push Button wykorzystam do obsługi zdarzeń dla danych. Będzie to szereg przycisków służących do uruchamiania funkcji obliczających różne algorytmy.
- Label będzie służył do podpisania danych grup z przyciskami.



Stworzenia tego okna wykorzystałem pięć GridLayoutów. Do każdego z nich wstawiłem różne przyciski. Do layoutu Classification wstawiłem ComboBox do wyboru klasyfikatora do mojego algorytmu oraz przycisk który będzie uruchamiał zdarzenie wywołujące funkcję klasyfikacji. Do layoutu Clusterisation wstawiłem przycisk wywołujący funkcję która będzie uruchamiać algorytm k-sąsiadów grupujący dane kolumn zaznaczonych w tabeli. Do layoutu Coorelation wstawiłem dwa ComboBoxy w których będzie mogli wybrać dwa atrybuty do obliczenia korelacji pomiędzy nimi. W layoucie podpisanym jako Plots wstawiłem trzy przyciski do generowania trzech różnych wykresów, każdy z nich będzie generował wykres dla zaznaczonych kolumn danych w tabeli. W piątym layoucie o nazwie Stats wstawione jest 5 przycisków które wywołują funkcje obliczające dane statystyczne dla zaznaczonych kolumn. Będzie możliwość zaznaczenia kilku z nich za pomocą checkboxów i policzenia zaznaczonych funkcji statystycznych za pomocą jednego przycisku Calculate Selected. Jest tu również przycisk Clear który będzie czyścił okno wyświetlające obliczenia. Na środku są dwa TextBrowsery . Jeden do wyświetlania obliczeń i drugi do wyświetlania informacji o przyciskach. U góry widać TableView, natomiast w lewym górnym rogu jest pasek menu z dodatkowymi funkcjami. Będzie tam wczytywanie oraz zapisywanie plików dla różnych użytkowników, wpisywanie nagłówek do tabeli, generowanie raportu PDF, zmiana motywu aplikacji oraz wyszukiwanie danych w tabeli. Następnie stworzyłem okno logowania. Jest ono mniej złożone niż główne okno aplikacji.



Najpierw wstawiłem dwa GridLayouty. W dolnym uporządkowałem przyciski uruchamiające funkcje logowania jako użytkownik lub administrator, logowania jako gość oraz rejestracji. Do górnego wstawiłem QLineEdit w które wpisuje się login oraz hasło. Następnie stworzyłem okno rejestracji.



Wstawiłem gridlayout w którym umieściłem QLineEdit do wprowadzania danych niezbędnych do rejestracji oraz umieściłem swobodnie checkbox do pokazywania hasła oraz przycisk wywołujący funkcję rejestracji. Okna zapisałem w rozszerzeniu .ui wraz z innymi plikami projektu.

6.1.4. Konwertowanie plików .ui do Python

Aby ułatwić sobie pracę przy kodowaniu okien postanowiłem zmienić reprezentację plików na .py.

Do konwersji plików .ui które reprezentują okna za pomocą języka XML wykorzystałem pyuic6. Jest to narzędzie wiersza poleceń dostarczane wraz z biblioteką PyQt6, służące do konwertowania plików interfejsu użytkownika (UI) utworzonych w programie Qt Designer (lub innym narzędziu do projektowania interfejsów opartym na Qt) na kod źródłowy w języku

Pythona. Ten wygenerowany kod Pythona będzie następnie używany do tworzenia interfejsu użytkownika w aplikacji.

Aby przekonwertować plik należy wykonać polecenie:

```
pyuic6 -x main.ui -o main.py
```

-x oznacza że plik może zawierać dodatkowy kod który wykona się podczas importowania pliku, natomiast -o oznacza output, określa do jakiego pliku ma zostać zapisany wygenerowany kod.

6.2. Kodowanie komunikacji między oknami.

Mając gotowe pliki okien interfejsu pierwszą rzeczą jaką trzeba zakodować jest przechodzenie między danymi oknami. Pierwszą podstawową rzeczą, którą trzeba zrobić jest stworzenie instancji klasy QApplication z modułu QtWidgets z biblioteki PyQt6.

```
app = QtWidgets.QApplication(sys.argv)
```

- QtWidgets: Jest to moduł w bibliotece PyQt6, który zawiera klasy i funkcje związane z elementami interfejsu użytkownika (UI) w Qt.
- QApplication: Jest to główna klasa reprezentująca aplikację Qt. Tworzenie obiektu tej klasy jest konieczne, aby uruchomić aplikację PyQt. QApplication zarządza zasobami systemowymi i obsługuje zdarzenia, takie jak kliknięcia myszy, klawisze klawiatury itp.
- sys.argv: Jest to lista argumentów przekazywanych do programu wierszem poleceń. W przypadku aplikacji PyQt, sys.argv zawiera ścieżkę do pliku wykonywalnego oraz inne argumenty, które mogą być przekazywane do aplikacji przy jej uruchamianiu. Należy zaimportować do klasy głównej klasy okien z ich plików.

```
from main_ui import Ui_MainWindow
```

```
from LoginWidget import LoginWidget
```

```
from SignUpWidget import SignUpWidget
```

Pozwoli to stworzyć obiekty tych klas które będą naszymi oknami wyświetlanymi w czasie działania aplikacji.

```
sign_up_window = SignUpWidget()
```

```
login_window = LoginWidget(sign_up_window)
```

```
login_window.show()
```

```
sys.exit(app.exec())
```

```
-----
```

```
main_window = MainWindow()
```

```
main_window.show() # Pokaż główne okno
```

Nie jest to cały kod odpowiedzialny za komunikację okien. Jest to wybrana część kodu która ukazuje tworzenie obiektów okien. Każde okno ma swoją klasę w której okodawne są komponenty odpowiedzialne za funkcjonalności danego okna oraz specjalna metoda `def __init__` która uruchamia część kodu podczas tworzenia danego obiektu, jest konstruktorem, będzie nam ona szczególnie przydatna w kolejnych etapach kodowania naszych okien.

6.3. Kodowanie przycisków odpowiedzialnych za funkcjonalności systemu.

6.3.1. Programowanie metod inicjalizujących klasy okien

W pierwszej kolejności musimy uzupełnić metodę init dla klasy głównego okna. Klasa musi dostać informacje kto zalogował się do systemu żeby określić dostępność funkcjonalności. Jeśli zaloguje się gość część funkcjonalności zostanie ograniczona w tym segmencie kodu poprzez nieuwidocznienie poszczególnych elementów okna głównego. Kolejnym istotnym aspektem jest stworzenie obiektu DataFrame, przypisanie danych do tabeli poprzez stworzenie modelu danych oraz stworzenie obiektu DataFrame do którego również wprowadza się dane, ponieważ jest to struktura danych która oferuje dużo funkcji do operowania na danych oraz jest ona kompatybilna z wieloma innymi bibliotekami. Wstawiamy również odpowiednie dane z nagłówek do comboboxów. Uruchamiamy wątek odpowiedzialny za przysyłanie edytowanych danych z tabeli do DataFrame. Następnie programujemy zdarzenia przycisków oraz menu.

```
# obsługa paska menu
self.actionCSV_load.triggered.connect(
    lambda: Functions.load_CSV_file(self.comboBoxAtrybut1, self.comboBoxAtrybut2, self.comboBoxAtrybutClass,
    self.tableView))
self.actionCSV_save.triggered.connect(lambda: Functions.save_to_CSV())
self.actionJSON_load.triggered.connect(
    lambda: Functions.load_JSON_file(self.comboBoxAtrybut1, self.comboBoxAtrybut2, self.comboBoxAtrybutClass,
    self.tableView))
self.actionJSON_save.triggered.connect(lambda: Functions.save_to_JSON())
self.actionWprowadzNagl.triggered.connect(
    lambda: Functions.add_headers_manually(self.comboBoxAtrybut1, self.comboBoxAtrybut2,
    self.comboBoxAtrybutClass, self.tableView))
self.actionGenerateResultsPDF.triggered.connect(lambda: Functions.generate_pdf(self.textBrowser))
self.actionCiemny.triggered.connect(lambda: Functions.change_to_darkmode(self, self.tableView, self.textBrowserInfo))
self.actionJasny.triggered.connect(lambda: Functions.change_to_lightmode(self, self.tableView))
self.actionWyszukaj.triggered.connect(lambda: self.search())
```

Każde zdarzenie zaprogramowane jest w następującym schemacie
self.nazwa_akcji.triggered.connect(lambda:funkcja(argumenty))
self oznacza dostęp do obiektu który został zainicjalizowany w tym kodzie, w tym przypadku będzie to obiekt okna głównego. Okno główne ma menu, każdy przycisk menu ma własną nazwę np. actionCSV_load to przycisk w menu do wczytywania danych z pliku CSV. Słowo kluczowe „triggered” oznacza, że zostało uruchomione zdarzenie np. Naciśnięcie przycisku w menu. Słowo kluczowe „connect” określa metodę która służy do łączenia zdarzenia danego elementu interfejsu z odpowiednią funkcją w skrypcie. Następnie wywoływana jest funkcja anonimowa za pomocą słowa kluczowego lambda. Zanim zacząłem używać tych funkcji natrafiłem na problem, ponieważ funkcje uruchamiały się podczas inicjalizacji przycisków w aplikacji, a nie po naciśnięciu przycisku co było błędem. Udało mi się rozwiązać ten błąd za pomocą opóźnienia wywoływania funkcji przez użycie lambdy. Niektóre funkcje przyjmują jako argumenty obiekty komponentów interfejsu graficznego, co pozwala danym funkcjom je modyfikować.

```
# przyciski statystyczne: Obsługa:
self.pushButtonMinimum.clicked.connect(lambda: Functions.calculate_minimum(self.textBrowser, self.tableView))
self.pushButtonMaximum.clicked.connect(lambda: Functions.calculate_maximum(self.textBrowser, self.tableView))
self.pushButtonClear.clicked.connect(lambda: self.textBrowser.clear())
self.pushButtonMean.clicked.connect(lambda: Functions.calculate_mean(self.textBrowser, self.tableView))
self.pushButtonStd.clicked.connect(lambda: Functions.calculate_std(self.textBrowser, self.tableView))
self.pushButtonMedian.clicked.connect(lambda: Functions.calculate_median(self.textBrowser, self.tableView))
self.pushButtonDystrybucja.clicked.connect(lambda: Functions.generate_distribution_plot(self.tableView))
self.pushButtonKoorelacja.clicked.connect(
    lambda: Functions.calculate_coorelation(self.comboBoxAtrybut1, self.comboBoxAtrybut2, self.textBrowser,
    self.tableView))
self.pushButtonCalcChecked.clicked.connect(
    lambda: Functions.calculate_checked_stats(self.checkBoxMin, self.checkBoxMax, self.checkBoxStd, self.checkBoxMdn,
    self.checkBoxMean, self.textBrowser, self.tableView))
```

W analogiczny sposób zakodowałem przyciski odpowiedzialne za obliczenia statystyczne oraz algorytmy uczenia maszynowego. Następnie trzeba zaprogramować podpięte funkcje, ponieważ na tym etapie tworzenia aplikacji są one jeszcze puste, więc po wywołaniu ich nic się nie stanie.

6.3.2. Kodowanie funkcji odpowiedzialnych za obliczenia matematyczne

- Funkcja wyznaczająca minimum dla danej kolumny

```
2 usages (1 dynamic)  Krzysztof Sendra +1
1 def calculate_minimum(QtTextBrowser, QtTableView): #git
2     try:
3         for column in selected_columns:
4             # Oblicz minimum z wybranej kolumny
5             minimum = df[column].min()
6
7             # Konwertuj wartość na string
8             minimum_str = str(minimum)
9
10            # Pobierz tekst nagłówka kolumny z QtTableView
11            header_text = QtTableView.model().headerData(column, Qt.Orientation.Horizontal)
12
13            # Wyświetl wartość w QTextBrowser
14            QtTextBrowser.append(f"Minimum from column {header_text} is: {minimum_str}")
15        except Exception as e:
16            QtTextBrowser.append(f"Error occurred: {str(e)}")
17
```

Nazwa funkcji to `calculate_minimum`, posiada ona dwa argumenty `QtTextBrowser` oraz `QtTableView` co oznacza że przyjmuje ona do siebie dwa obiekty widżetów: Tabeli oraz ekranu do wyświetlania obliczeń. Do obliczenia wartości wykorzystujemy funkcję `.min()`, która jest wbudowana w Pythonie i można ją użyć na każdej iterowalnej strukturze danych. W tym przypadku oblicza się ją na podstawie znanego kolumny w tabeli i zapisuje do zmiennej `minimum`. Następnie wartość konwertowana jest jako ciąg znaków tekstowych i wyświetlana w obiekcie ekranu za pomocą metody `append()`. Dodatkowo cała funkcja zabezpieczona jest by po wystąpieniu błędu nie zawiesiła programu tylko wypisał go na ekranie.

- Funkcja wyznaczająca maksimum dla danej kolumny

```

228 def calculate_maximum(QtTextBrowser, QtTableView): #git
229     try:
230         for column in selected_columns:
231             # Oblicz maksimum z wybranej kolumny
232             maximum = df[column].max()
233
234             # Konwertuj wartość na string
235             maximum_str = str(maximum)
236
237             # Pobierz tekst nagłówka kolumny z QtTableView
238             header_text = QtTableView.model().headerData(column, Qt.Orientation.Horizontal)
239
240             # Wyświetl wartość w QTextBrowser
241             QTextBrowser.append(f"Maximum from column {header_text} is: {maximum_str}")
242     except Exception as e:
243         QTextBrowser.append(f"Error occurred: {str(e)}")

```

Funkcja odpowiedzialna za wyznaczenie maksimum jest analogicznie skonstruowana do tej wyznaczającej minimum. Wbudowana funkcja Python `max()` działa bardzo podobnie do funkcji `min()`. Wyznacza maksymalną wartość z iterowalnej struktury danych. Wynik wypisywany jest na ekranie do wyświetlania wyników.

- Funkcja wyznaczająca medianę

```

245 def calculate_median(QtTextBrowser, QtTableView): #git
246     try:
247         for column in selected_columns:
248             # Oblicz medianę z wybranej kolumny
249             median = df[column].median()
250
251             # Konwertuj wartość na string
252             median_str = str(median)
253
254             # Pobierz tekst nagłówka kolumny z QtTableView
255             header_text = QtTableView.model().headerData(column, Qt.Orientation.Horizontal)
256
257             # Wyświetl wartość w QTextBrowser
258             QTextBrowser.append(f"Median from column {header_text} is: {median_str}")
259     except Exception as e:
260         QTextBrowser.append(f"Error occurred: {str(e)}")

```

Mediana to środkowa wartość z tablicy liczb lub średnia arytmetyczna dwóch środkowych jeśli liczba elementów jest nieparzysta. W tym przypadku również jest to funkcja wbudowana w Pythonie. Wyznaczoną medianę wypisujemy na ekranie do obliczeń.

- Funkcja wyznaczająca odchylenie standardowe

```

262 def calculate_std(QtTextBrowser, QtTableView): #git
263     try:
264         for column in selected_columns:
265             # Oblicz odchylenie standardowe z wybranej kolumny
266             std = df[column].std()
267
268             # Konwertuj wartość na string
269             std_str = str(std)
270
271             # Pobierz tekst nagłówka kolumny z QtTableView
272             header_text = QtTableView.model().headerData(column, Qt.Orientation.Horizontal)
273
274             # Wyświetl wartość w QTextBrowser
275             QTextBrowser.append(f"Standard deviation from column {header_text} is: {std_str}")
276     except Exception as e:
277         QTextBrowser.append(f"Error occurred: {str(e)}")

```

- Funkcja wyznaczająca średnią arytmetyczną

```

279 def calculate_mean(QtTextBrowser, QtTableView): #git
280     try:
281         for column in selected_columns:
282             # Oblicz średnią z wybranej kolumny
283             mean = (df[column].mean())
284
285             # Konwertuj wartość na string
286             mean_str = str(mean)
287
288             # Pobierz tekst nagłówka kolumny z QtTableView
289             header_text = QtTableView.model().headerData(column, Qt.Orientation.Horizontal)
290
291             # Wyświetl wartość w QtTextBrowser
292             QtTextBrowser.append(f"Mean from column {header_text} is: {mean_str}")
293     except Exception as e:
294         QtTextBrowser.append(f"Error occurred: {str(e)}")
295

```

- Funkcja wyznaczająca korelację dla dwóch wybranych atrybutów

```

296 def calculate_coorelation(comboBoxAtrybut1, comboBoxAtrybut2, PyQtTextBrowser, tableView):
297     model = tableView.model()
298     # pobierz indeks wybranej kolumny z comboBoxAtrybut1
299     column1_index = comboBoxAtrybut1.currentIndex() + 1
300     # pobierz indeks wybranej kolumny z comboBoxAtrybut2
301     column2_index = comboBoxAtrybut2.currentIndex() + 1
302     # pobierz dane z wybranej kolumny 1
303     column1_data = [model.data(model.index(row, column1_index)) for row in range(model.rowCount())]
304     # pobierz dane z wybranej kolumny 2
305     column2_data = [model.data(model.index(row, column2_index)) for row in range(model.rowCount())]
306     # utwórz nowy obiekt DataFrame z pobranych danych
307     data = pd.DataFrame(
308         {comboBoxAtrybut1.currentText(): column1_data, comboBoxAtrybut2.currentText(): column2_data})
309     # data = code_data(data)
310     data = data.apply(pd.to_numeric)
311     correlation_matrix = data[[comboBoxAtrybut1.currentText(), comboBoxAtrybut2.currentText()]].corr()
312     correlation_coefficient = correlation_matrix.iloc[0, 1]
313     PyQtTextBrowser.append(f"Koorelacja między tymi atrybutami wynosi: " + str(correlation_coefficient))

```

- Funkcja wyznaczająca zaznaczone statystyki

```

Usage  Krzys3n
316 def calculate_checked_stats(checkBoxMin, checkBoxMax, checkBoxStd, checkBoxMdn, checkBoxMean, textBrowser, tableView):
317     if (checkBoxMin.isChecked() == True):
318         calculate_minimum(textBrowser, tableView)
319     if (checkBoxMax.isChecked() == True):
320         calculate_maximum(textBrowser, tableView)
321     if (checkBoxStd.isChecked() == True):
322         calculate_std(textBrowser, tableView)
323     if (checkBoxMdn.isChecked() == True):
324         calculate_median(textBrowser, tableView)
325     if (checkBoxMean.isChecked() == True):
326         calculate_mean(textBrowser, tableView)
327

```

6.3.3. Kodowanie generowania wykresów

- Funkcja generująca wykres zliczeń dla zaznaczonych kolumn


```

330 def generate_comparison_plot(tableView): #git
331     model = tableView.model()
332     data = pd.DataFrame() # Utwórz pusty obiekt DataFrame
333
334     for column_index in selected_columns:
335         # Pobierz nazwę wybranej kolumny z QTableView
336         column_name = model.headerData(column_index, Qt.Orientation.Horizontal)
337         # Pobierz dane z wybranej kolumny
338         column_data = [model.data(model.index(row, column_index)) for row in range(model.rowCount())]
339         # Dodaj dane do obiektu DataFrame
340         data[column_name] = column_data
341
342     # Konwertuj dane na typ numeryczny
343     data = data.apply(pd.to_numeric, errors='coerce')
344     counts = data.apply(pd.Series.value_counts).fillna(0)
345     counts.plot(kind='bar')
346     plt.xlabel('Atrybut')
347     plt.ylabel('Suma')
348     # Skonstruuj tytuł wykresu na podstawie nazw zaznaczonych atrybutów
349     title = "Zliczenia "
350
351     for column_name in data.columns:
352         title += str(column_name) + " + "
353     title = title[:-3] # Usuń ostatni znak "+" i spację
354     plt.title(title)
355     plt.show()

```

- Funkcja generująca wykres dystrybucji dla zaznaczonych kolumn

```

386 def generate_distribution_plot(tableView): #git
387     model = tableView.model()
388     data = pd.DataFrame() # Utwórz pusty obiekt DataFrame
389
390     for column_index in selected_columns:
391         # Pobierz nazwę wybranej kolumny z QTableView
392         column_name = model.headerData(column_index, Qt.Orientation.Horizontal)
393         # Pobierz dane z wybranej kolumny
394         column_data = [model.data(model.index(row, column_index)) for row in range(model.rowCount())]
395         # Dodaj dane do obiektu DataFrame
396         data[column_name] = column_data
397
398     # Konwertuj dane na typ numeryczny
399     data = data.apply(pd.to_numeric, errors='coerce')
400     # Sumuj wartości dla każdej kolumny
401     counts = data.sum()
402     # Wygeneruj wykres kołowy
403     plt.pie(counts.values, labels=counts.index, autopct='%1.1f%%')
404     plt.axis('equal')
405
406     # Skonstruuj tytuł wykresu na podstawie nazw zaznaczonych atrybutów
407     title = "Rozkład "
408     for column_name in data.columns:
409         title += str(column_name) + " + "
410     title = title[:-3] # Usuń ostatni znak "+" i spację
411     plt.title(title)
412     plt.show()
413

```

- Funkcja generująca heatmapę korelacji dla zaznaczonych kolumn

```

357 def generate_correlation_heatmap(tableView): #git
358     model = tableView.model()
359     data = pd.DataFrame() # Utwórz pusty obiekt DataFrame
360
361     for column_index in selected_columns:
362         # Pobierz nazwę wybranej kolumny z QTableView
363         column_name = model.headerData(column_index, Qt.Orientation.Horizontal)
364         # Pobierz dane z wybranej kolumny
365         column_data = [model.data(model.index(row, column_index)) for row in range(model.rowCount())]
366         # Dodaj dane do obiektu DataFrame

```


6.3.4. Kodowanie wczytywania oraz zapisywania plików

- Funkcja wyczytująca dane z formatu CSV

```
1 usage  ± Krzysztof Sendera +1 *
415 def load_CSV_file(comboBoxAtrybut1, comboBoxAtrybut2, comboBoxAtrybutClass, tableView):
416     global df
417     global df_with_labels
418     global labels
419     msg_box = QMessageBox()
420     msg_box.setWindowTitle("Wczytywanie pliku CSV")
421     msg_box.setText("Czy chcesz wczytać plik z nazwami kolumn?")
422     yes_button = msg_box.addButton(QMessageBox.StandardButton.Yes)
423     yes_button.setText("Tak, z nazwami")
424     msg_box.addButton(QMessageBox.StandardButton.No).setText("Nie, bez nazw")
425     cn_button = msg_box.addButton(QMessageBox.StandardButton.Cancel)
426     cn_button.setVisible(False)
427     msg_box.setDefaultButton(yes_button)
428     reply = msg_box.exec()
429
430     if reply == 16384:
431         header = 0
432     elif reply == 65536:
433         header = None
434     else:
435         return
436
437     current_dir = os.getcwd()
438     filename, _ = QFileDialog.getOpenFileName(None, "Wybierz plik CSV lub DATA", current_dir,
439                                             "All files (*);;Pliki CSV (*.csv);;Pliki DATA (*.data)")
440
441     if not filename:
442         # Wyjście z funkcji, jeśli nie został wybrany plik
443         return
444
445     # Wczytaj dane z pliku CSV do obiektu DataFrame z biblioteki Pandas
446     df = pd.read_csv(filename, header=header)
447     df_with_labels = df.copy()
448     labels = df.columns.tolist()
449
450     # Wyświetl dane za pomocą funkcji print
451
452     model = QStandardItemModel(df.shape[0], df.shape[1])
453     headers = list(df.columns)
454     if reply == 16384:
455         model.setHorizontalHeaderLabels(headers)
456
457     for row in range(df.shape[0]):
458         for column in range(df.shape[1]):
459             item = QStandardItem(str(df.iloc[row, column]))
460             model.setItem(row, column, item)
461
462     tableView.setModel(model)
463
464     # wczytywanie nazw kolumn do comboboxów
465     column_names = df.columns.tolist()[1:]
466     comboBoxAtrybut1.clear()
467     comboBoxAtrybut2.clear()
468     comboBoxAtrybutClass.clear()
469     comboBoxAtrybut1.addItem(str(x) for x in column_names)
470     comboBoxAtrybut2.addItem(str(x) for x in column_names)
471     comboBoxAtrybutClass.addItem(str(x) for x in column_names)
472
473     # Liczba kolumn minus 1
474     x = len(df.columns) - 1
475     # Ustawienie nazw kolumn na liczby od 0 do x
476     df.columns = list(range(x + 1))
477
478     tableView.setSelectionMode(QTableView.SelectionMode.ExtendedSelection)
479     selection_model = tableView.selectionModel()
480     selection_model.selectionChanged.connect(lambda: handle_selection_changed(tableView))
```

- Funkcja wczytująca dane z formatu JSON

```

481 def load_JSON_file(comboBoxAtrybut1, comboBoxAtrybut2, comboBoxAtrybutClass, tableView):
482     global df
483     global df_with_labels
484     current_dir = os.getcwd()
485     filename, _ = QFileDialog.getOpenFileName(None, "Wybierz plik JSON", current_dir, "Pliki JSON (*.json)")
486
487     if not filename:
488         # Wyjście z funkcji, jeśli nie został wybrany plik
489         return
490
491     # Wczytaj dane z pliku JSON do obiektu DataFrame z biblioteki Pandas
492     with open(filename, 'r') as json_file:
493         data = json.load(json_file)
494
495     df = pd.DataFrame(data)
496     df_with_labels = df.copy()
497
498     # Wyświetl dane w tabeli
499     model = QStandardItemModel(df.shape[0], df.shape[1])
500     headers = list(df.columns)
501     model.setHorizontalHeaderLabels(headers)
502
503     for row in range(df.shape[0]):
504         for column in range(df.shape[1]):
505             item = QStandardItem(str(df.iloc[row, column]))
506             model.setItem(row, column, item)
507
508     tableView.setModel(model)
509
510     # Wczytywanie nazw kolumn do comboboxów
511     column_names = df.columns.tolist()
512     comboBoxAtrybut1.clear()
513     comboBoxAtrybut2.clear()
514     comboBoxAtrybutClass.clear()
515     comboBoxAtrybut1.addItem(column_names)
516     comboBoxAtrybut2.addItem(column_names)
517     comboBoxAtrybutClass.addItem(column_names)
518
519     tableView.setSelectionMode(QTableView.SelectionMode.ExtendedSelection)
520     selection_model = tableView.selectionModel()
521     selection_model.selectionChanged.connect(lambda: handle_selection_changed(tableView))

```

- Funkcja zapisująca dane do formatu CSV

```

1 usage  ± Krzysztof Sendera +1 *
524 def save_to_CSV():
525     # Okno dialogowe z pytaniem o zapisanie z nazwami kolumn lub bez
526     msg_box = QMessageBox()
527     msg_box.setWindowTitle("Zapisywanie pliku CSV")
528     msg_box.setText("Czy chcesz zapisać plik z nazwami kolumn?")
529
530     yes_button = msg_box.addButton(QMessageBox.StandardButton.Yes)
531     yes_button.setText("Tak")
532     msg_box.addButton(QMessageBox.StandardButton.No).setText("Nie")
533     cn_button = msg_box.addButton(QMessageBox.StandardButton.Cancel)
534     cn_button.setVisible(False)
535
536     msg_box.setDefaultButton(yes_button)
537
538     reply = msg_box.exec()
539
540     if reply == 16384:
541         header = True
542     elif reply == 65536:
543         header = False
544     else:
545         return
546
547     current_dir = os.getcwd()
548     filename, _ = QFileDialog.getSaveFileName(None, "Zapisz plik CSV", current_dir, "Pliki CSV (*.csv)")
549     if filename:
550         df_with_labels.to_csv(filename, index=False, header=header)

```

- Funkcja zapisująca dane do formatu JSON

```

1 usage  ± Krzys3n
554 def save_to_JSON():
555     current_dir = os.getcwd()
556     filename, _ = QFileDialog.getSaveFileName(None, "Zapisz plik JSON", current_dir, "Pliki JSON (*.json)")
557
558     if filename:
559         # Zakładam, że masz DataFrame o nazwie df_with_labels, które chcesz zapisać do pliku JSON.
560         # Jeśli używasz innej zmiennej, zmień ją w odpowiednim miejscu.
561         data_to_save = df.to_dict(orient='records')
562
563         with open(filename, 'w') as json_file:
564             json.dump(data_to_save, json_file, indent=4)
565

```

6.3.5. Kodowanie generowania raportu PDF

```

671 def generate_pdf(QtTextBrowser):
672     # Wyświetl okno dialogowe do wyboru miejsca zapisu pliku
673     file_dialog = QFileDialog()
674     file_dialog.setWindowTitle("Save PDF")
675     file_dialog.setFileMode(QFileDialog.FileMode.AnyFile)
676     file_dialog.setAcceptMode(QFileDialog.AcceptMode.AcceptSave)
677     file_dialog.setNameFilter("PDF Files (*.pdf)")
678     file_dialog.setDefaultSuffix("pdf")
679
680     if file_dialog.exec() == QFileDialog.DialogCode.Accepted:
681         file_path = file_dialog.selectedFiles()[0]
682     else:
683         return # Użytkownik anulował wybór pliku
684
685     if file_path:
686         # Pobierz tekst z QTextBrowser
687         text = QtTextBrowser.toPlainText()
688         # Utwórz nowy obiekt Canvas
689         c = canvas.Canvas(file_path)
690         # Ustaw czcionkę i rozmiar tekstu
691         c.setFont("Helvetica", 12)
692         # Wypisz tekst w pliku PDF
693         lines = text.split("\n")
694         y = 800 # Wysokość początkowa
695         for line in lines:
696             c.drawString(50, y, line)
697             y -= 20 # Zmniejsz wysokość dla kolejnej linii
698         # Zamknij plik PDF
699         c.save()

```

6.3.6. Kodowanie wyświetlania informacji o przyciskach po najechaniu na nie kursorem myszy

```
702 ## Sekcja wyświetlania informacji:
703 # wyświetlanie informacji o przyciskach:
1 usage  └ Krzys3n *
704 def switch_dictionary_buttons(button_name): #git
705     switcher = {
706         "pushButtonMinimum": "Calculates the minimum of selected columns data",
707         "pushButtonMaximum": "Calculates the maximum of selected columns data",
708         "pushButtonStd": "Calculates the standard deviation of selected columns data",
709         "pushButtonMedian": "Calculates the median of selected columns data",
710         "pushButtonMean": "Calculates the arithmetic average of selected columns data",
711         "pushButtonClear": "Clears the result window",
712         "pushButtonCalcChecked": "Designates selected descriptive statistics",
713         "pushButtonKorelacja": "Calculates the correlation of selected attributes and generates a correlation heatmap",
714         "pushButtonPorownaj": "Generates a comparison plot of selected columns data",
715         "pushButtonDystrybucja": "Generates a distribution plot of selected columns data",
716         "pushButtonHeatmap": "Generates a coorelation Heatmap of selected columns data",
717         "pushButtonClass": "Provides classification of selected attribute for selected data",
718         "pushButtonCluster": "Provides clusterization for selected clusters for selected data"
719     }
720
721     return switcher.get(button_name, "No information available for the button")
722
723
12 usages  └ Krzys3n
724 def on_button_enter(event: QEnterEvent, button_name: str, QTextBrowser): #git
725     info = switch_dictionary_buttons(button_name)
726     QTextBrowser.setText(info)
727     QTextBrowser.setAlignment(Qt.AlignmentFlag.AlignCenter)
728
```

6.3.7. Kodowanie stylów, które umożliwiają wybór motywu aplikacji

- Funkcja zmieniająca motyw na jasny

```
1 usage  └ Krzys3n *
803 def change_to_lightmode(window, table_view):
804     window.setStyleSheet("")
805     table_view.horizontalHeader().setVisible(True)
806     table_view.verticalHeader().setVisible(True)
807     table_view.setStyleSheet("""
808         background-color: #ffffff;
809         color: black;
810         font-family: Titillium;
811         font-size: 18px;
812
813     })
814
815
816     table_view.horizontalHeader().setStyleSheet("QHeaderView::section { background-color: #ffffff; color: black; }")
817     table_view.verticalHeader().setStyleSheet("QHeaderView::section { background-color: #ffffff; color: black; }")
818
```

- Funkcja zmieniająca motyw na ciemny

```
781 ## Sekcja styli
1 usage  └ Krzys3n
782 def change_to_darkmode(window, table_view, textBrowserInfo):
783     window.setStyleSheet("""
784         background-color: #262626;
785         color: white;
786         font-family: Titillium;
787         font-size: 14px;
788
789     })
790
791     table_view.setStyleSheet("""
792         background-color: #262626;
793         color: white;
794         font-family: Titillium;
795         font-size: 18px;
796
797     })
798
799     table_view.horizontalHeader().setStyleSheet("QHeaderView::section { background-color: #262626; color: white; }")
800     table_view.verticalHeader().setStyleSheet("QHeaderView::section { background-color: #262626; color: white; }")
801
```

6.4. Podłączenie projektu do bazy danych z użytkownikami

6.5. Eksportowanie aplikacji użytkowej.

Aplikację użytkową eksportujemy za pomocą oprogramowania pyinstaller. Jest to narzędzie służące do pakowania aplikacji napisanych w języku Python w jednym pliku wykonywalnym. Jest to narzędzie umożliwiające łatwe tworzenie samodzielnych plików wykonywalnych (.exe na systemach Windows, .app na systemach macOS, itp.) z kodu Źródłowego Pythona.

Główne zalety PyInstallera to:

- PyInstaller pozwala na utworzenie jednobazowych plików wykonywalnych, co oznacza, że cała aplikacja jest zawarta w jednym pliku, co ułatwia przenoszenie i udostępnianie aplikacji.
- Obsługa wielu platform: PyInstaller jest wieloplatformowy, co oznacza, że możesz używać go do pakowania aplikacji na różne systemy operacyjne, takie jak Windows, macOS i Linux.
- Obsługa zależności: PyInstaller jest w stanie automatycznie wykrywać i dołączać zależności Twojej aplikacji, takie jak moduły biblioteki standardowej Pythona oraz zewnętrzne biblioteki.
- Prosta obsługa: Proces używania PyInstallera jest stosunkowo prosty. Wystarczy uruchomić kilka poleceń w terminalu lub wierszu polecenia, aby uzyskać gotowy plik wykonywalny.

Moją aplikację wyeksportuję do jednego pliku za pomocą polecenia `pyinstaller -F MainWindow.py`

MainWindow.py to główny skrypt mojego projektu w którym zawarty jest kod uruchamiający całą aplikację.

Atrybut -F oznacza, że projekt ma zostać spakowany do jednego pliku. Można również spakować do folderu wraz z różnymi skryptami i bibliotekami używając tego polecenia bez atrybutu -F.

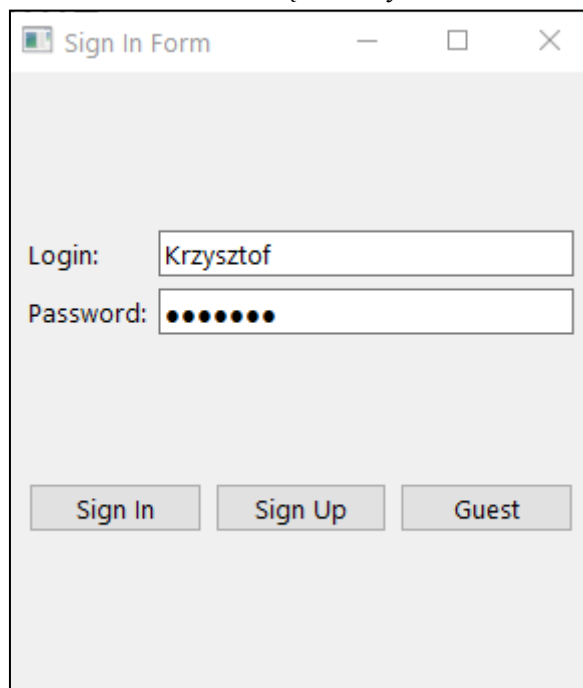
6.6. Kodowanie funkcji odpowiedzialnych za edycje, aktualizację oraz wypisywanie danych.

7. Sposób użytkowania aplikacji

Poniżej przedstawię przykładowy sposób użytkowania aplikacji.

7.1. Logowanie

Aby korzystać z systemu wymagane jest posiadanie konta o określonej roli, która umożliwia dostęp do określonych funkcji. Rola gościa pozwala nam korzystać z ograniczonej formy aplikacji, rola użytkownika z pełni jej funkcji, natomiast rola administratora pozwala nam również zarządzać użytkownikami.

A screenshot of a web application window titled "Sign In Form". The window has a light gray background and a standard Windows-style title bar with minimize, maximize, and close buttons. Inside the window, there are two input fields: "Login:" with the text "Krzysztof" and "Password:" with masked characters (dots). Below the input fields, there are three buttons: "Sign In", "Sign Up", and "Guest".

Wpisujemy dane logowania, następnie wybieramy przycisk Sign In aby się zalogować za pomocą wpisanych informacji. Jeśli chcemy po prostu wejść jako gość to logujemy się przyciskiem Guest. Zależnie od tego jaką rolę ma przypisaną użytkownik pokaże nam się ekran z odpowiednimi dostępnymi funkcjonalnościami. Zaznaczone funkcjonalności na rysunku poniżej zostaną opisane w kolejnych punktach.

The screenshot shows the PyNalise application window. At the top, there is a 'File' menu with options: Load From, Save to, Insert Headers, Generate PDF, Change theme, and Find. Below the menu is a table with columns: hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, breathes, and ver. The table contains data for four rows (4, 5, 6, 7). Below the table, there are several analysis tools: 'Classification: 7.7' with a dropdown for 'hair' and a 'Classificate' button; 'Clusterization: 7.8' with an 'Input cluster number' field and a 'Cluster' button; 'Correlation: 7.5' with a dropdown for 'hair' and a 'Count Coorelation' button; and 'Plots: 7.6' with buttons for 'Generate Count Plot', 'Generate Distribution Plot', and 'Generate Coorelation Plot'. At the bottom, there is a 'Stats: 7.5' section with checkboxes for 'Min', 'Max', 'Standard Deviation', 'Median', and 'Arithmetic Average', and a 'Calculate Selected' button.

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	ver
4	0	0	1	0	0	0	1	1	1	1	0
5	0	0	1	0	0	0	0	1	1	1	0
6	0	1	0	0	0	1	1	1	1	0	0
7	0	0	1	0	0	0	0	1	1	1	0

7.2. Rejestracja

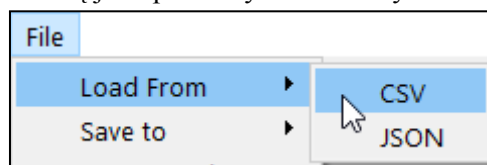
The screenshot shows a registration form titled 'Form'. It contains three input fields: 'E-mail:', 'Login:', and 'Password:'. There is a checkbox labeled 'Show Pass' next to the 'Login:' field. At the bottom of the form is a 'Sign Up' button.

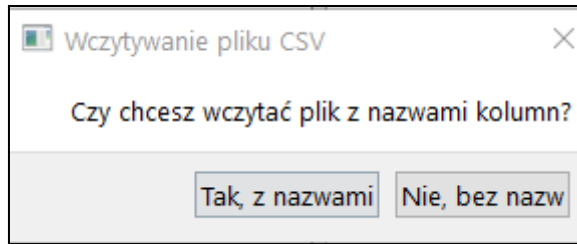
Aby się zarejestrować wpisujemy dane w odpowiednie pola. Po naciśnięciu przycisku Sign Up konto zostanie utworzone oraz dodane do systemu.

7.3. Wczytywanie Danych

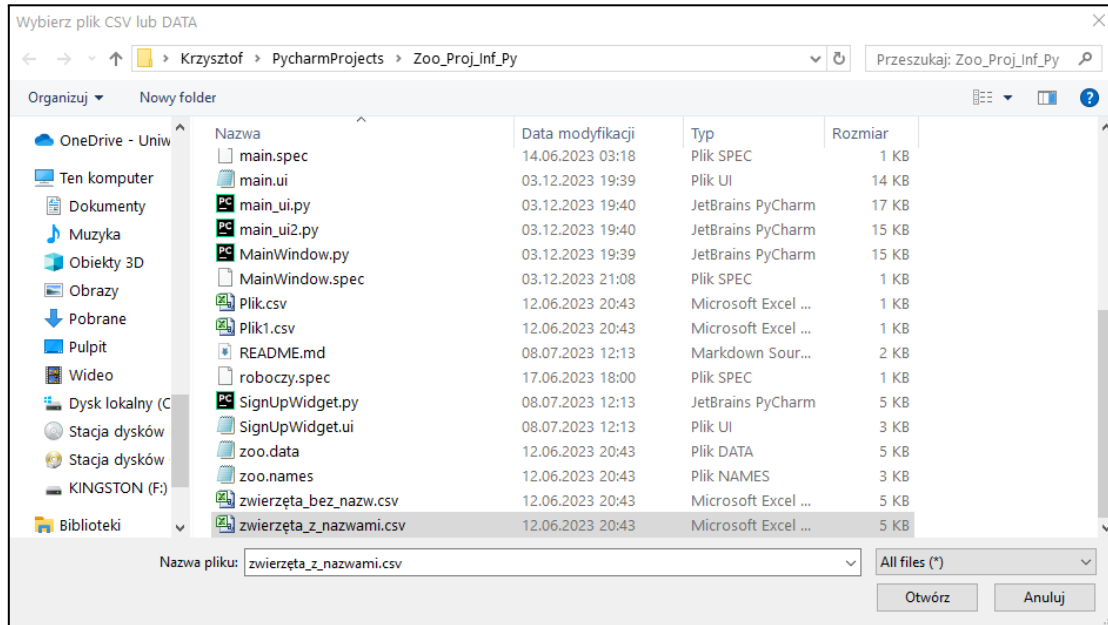
Dane wczytujemy wybierając odpowiednią opcję z rozwijanego menu w lewej górnej części ekranu.

Load From -> CSV wczytuje dane z pliku CSV z pierwszą linijką uznaną za nagłówek lub z pierwszą linijką uznaną jako pierwszy wiersz danych.

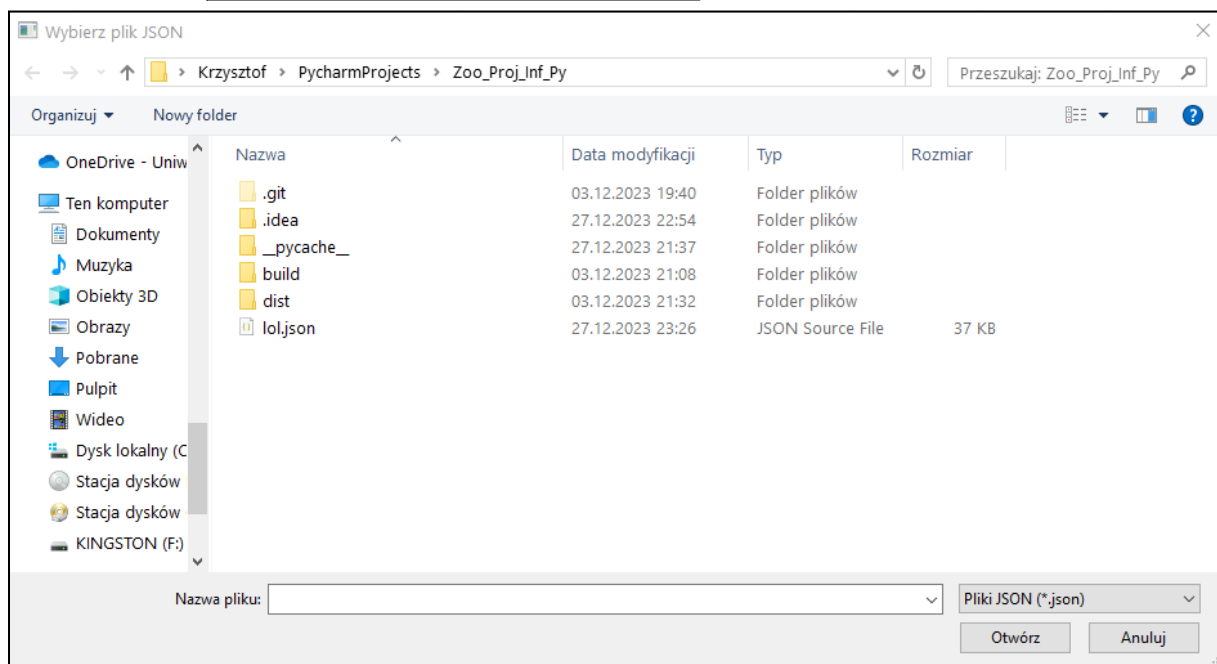
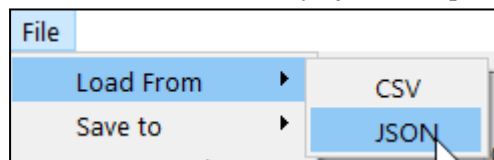




Wybieramy plik z okna wyboru pliku.

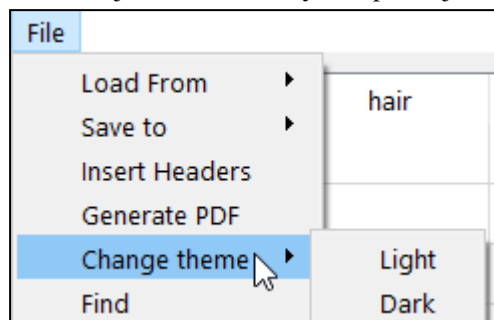


Load From -> JSON wczytuje dane z pliku o rozszerzeniu JSON.

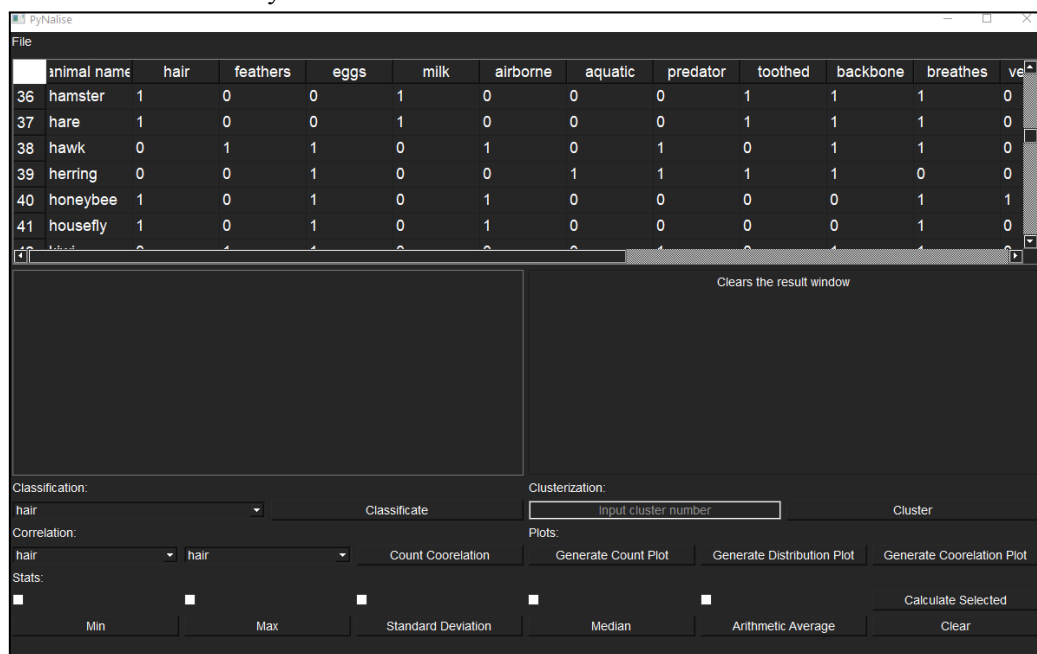


7.4. Konfiguracja stylu

Możliwa jest zmiana motywu aplikacji na ciemny lub jasny.



Aby to zrobić należy wejść w menu File, następnie wybrać opcję Change Theme->motyw



Dla przykładu w taki sposób prezentuje się ciemny motyw systemu.

7.5. Obliczanie statystyk

Do wyboru mamy 5 przycisków obliczających statystyki oraz jeden dodatkowy o nazwie Calculated Selected, który umożliwia obliczenie wszystkich zaznaczonych statystyk za pomocą checkboxów.

Minimum from column hair is: 0.0
Maximum from column hair is: 1.0
Median from column hair is: 0.0
Mean from column hair is: 0.4257425742573
Standard deviation from column hair is: 0.4969212141469781

Calculates the minimum of selected columns data

Classification:

hair

Classificate

Clusterization:

Input cluster number

Cluster

Correlation:

hair

hair

Count Coorelation

Plots:

Generate Count Plot

Generate Distribution Plot

Generate Coorelation Plot

Stats:

☐

☐

☒

☐

☐

Calculate Selected

Min

Max

Standard Deviation

Median

Arithmetic Average

Clear

Wybieramy przycisk statystyki która nas interesuje i wyświetla się ona w oknie wynikowym dla wybranej przez nas wcześniej kolumny. Statystyka obliczona zostaje dla wszystkich danych z wybranej aktualnie kolumny. Jeśli zaznaczymy wiele kolumn i wybierzemy przycisk Median to zostanie obliczone tyle median ile jest aktualnie zaznaczonych kolumn.

PyNalisse

File

	animal name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	vertebrate
1	aardvark	1	0	0	1	0	0	1	1	1	1	0
2	antelope	1	0	0	1	0	0	0	1	1	1	0
3	bass	0	0	1	0	0	1	1	1	1	0	0
4	bear	1	0	0	1	0	0	1	1	1	1	0
5	boar	1	0	0	1	0	0	1	1	1	1	0
6	buffalo	1	0	0	1	0	0	0	1	1	1	0

Median from column feathers is: 0.0
Median from column eggs is: 1.0
Median from column milk is: 0.0

Calculates the median of selected columns data

Classification:

hair

Classificate

Clusterization:

Input cluster number

Cluster

Correlation:

hair

hair

Count Coorelation

Plots:

Generate Count Plot

Generate Distribution Plot

Generate Coorelation Plot

Stats:

☐

☐

☐

☐

☐

Calculate Selected

Min

Max

Standard Deviation

Median

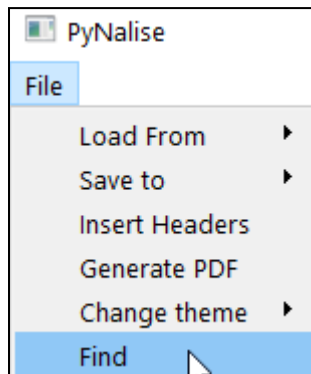
Arithmetic Average

Clear

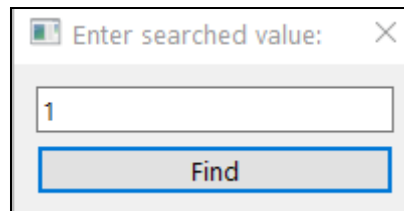
W tym przypadku została obliczona mediana dla kolumn z tabeli o nazwie “feathers”, “eggs” oraz “milk. Analogicznie działa reszta przycisków. Taki jest zalecany sposób poruszania się po danych w tym programie.

7.6. Wyszukiwanie

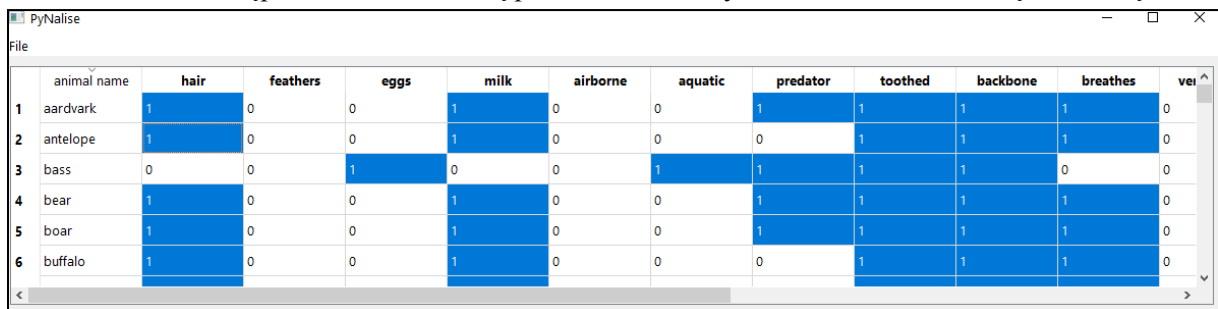
Wybieramy opcję Find z menu File aby wyszukać wartość we wszystkich komórkach tabeli.



Wpisujemy szukaną wartość do okienka dialogowego.



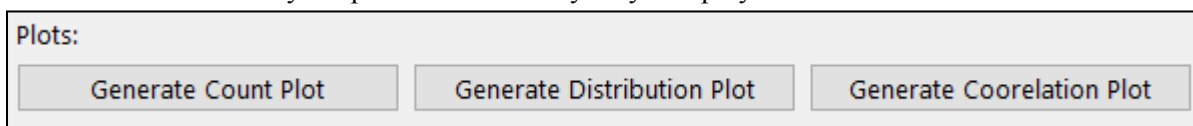
Następnie w tabeli zostaną podświetlone wszystkie komórki z szukaną wartością.

A screenshot of the PyNalise application window showing a table with 12 columns and 6 rows. The columns are: animal name, hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, breathes, and ver. The rows are numbered 1 to 6. The cells containing the value '1' are highlighted in blue. These cells are: (1, hair), (1, milk), (2, hair), (2, milk), (3, eggs), (3, aquatic), (4, hair), (4, milk), (5, hair), (5, milk), (6, hair), and (6, milk).

	animal name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	ver
1	aardvark	1	0	0	1	0	0	1	1	1	1	0
2	antelope	1	0	0	1	0	0	0	1	1	1	0
3	bass	0	0	1	0	0	1	1	1	1	0	0
4	bear	1	0	0	1	0	0	1	1	1	1	0
5	boar	1	0	0	1	0	0	1	1	1	1	0
6	buffalo	1	0	0	1	0	0	0	1	1	1	0

7.7. Generowanie Wykresów

Nasz system umożliwia generowanie trzech rodzajów wykresów, a mianowicie: dystrybucji, wyliczeń oraz heatmapy korelacji. Poniżej przedstawiam przykładowe generowanie każdego z nich dla atrybutów "feathers", "eggs" oraz "milk". Wystarczy zaznaczyć odpowiednie kolumny i wybrać przycisk.



Poniżej przedstawię przykładowe generowanie każdego z nich dla atrybutów "feathers", "eggs" oraz "milk". Zaznaczamy odpowiednie kolumny a następnie wybieramy przycisk.

PyNalise

File

	animal name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	ver
1	aardvark	1	0	0	1	0	0	1	1	1	1	0
2	antelope	1	0	0	1	0	0	0	1	1	1	0
3	bass	0	0	1	0	0	1	1	1	1	0	0
4	bear	1	0	0	1	0	0	1	1	1	1	0
5	boar	1	0	0	1	0	0	1	1	1	1	0
6	buffalo	1	0	0	1	0	0	0	1	1	1	0

The name of the animal

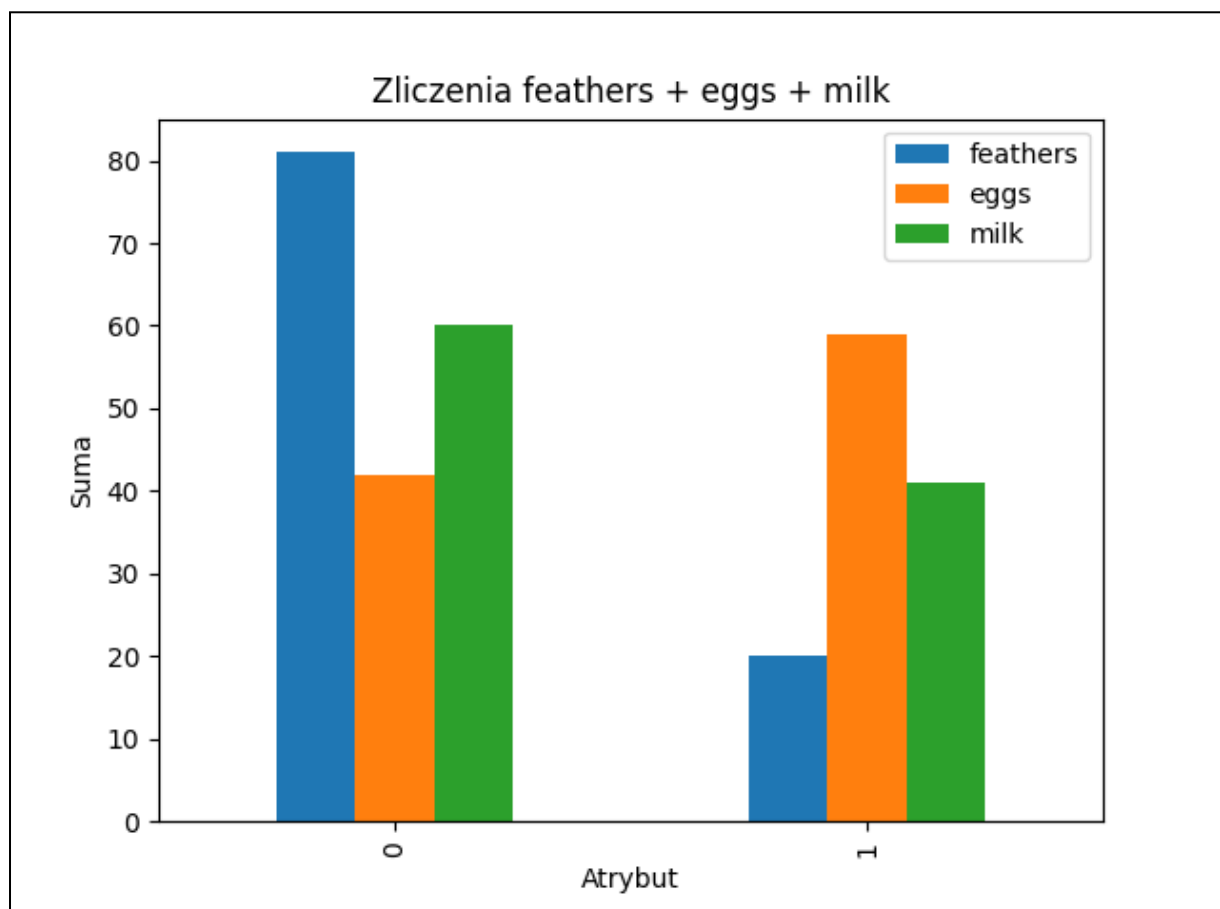
Classification:

Clusterization:

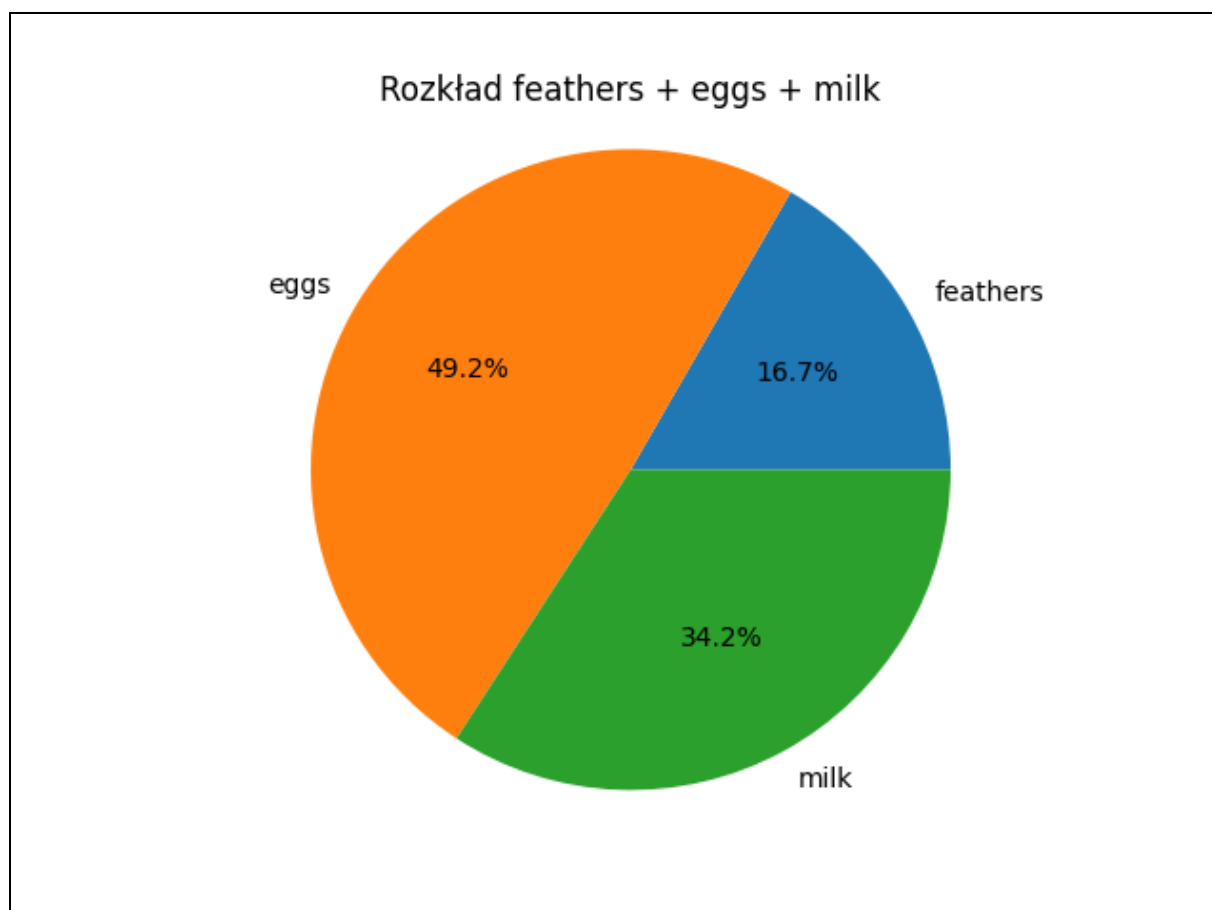
Correlation:

Plots:

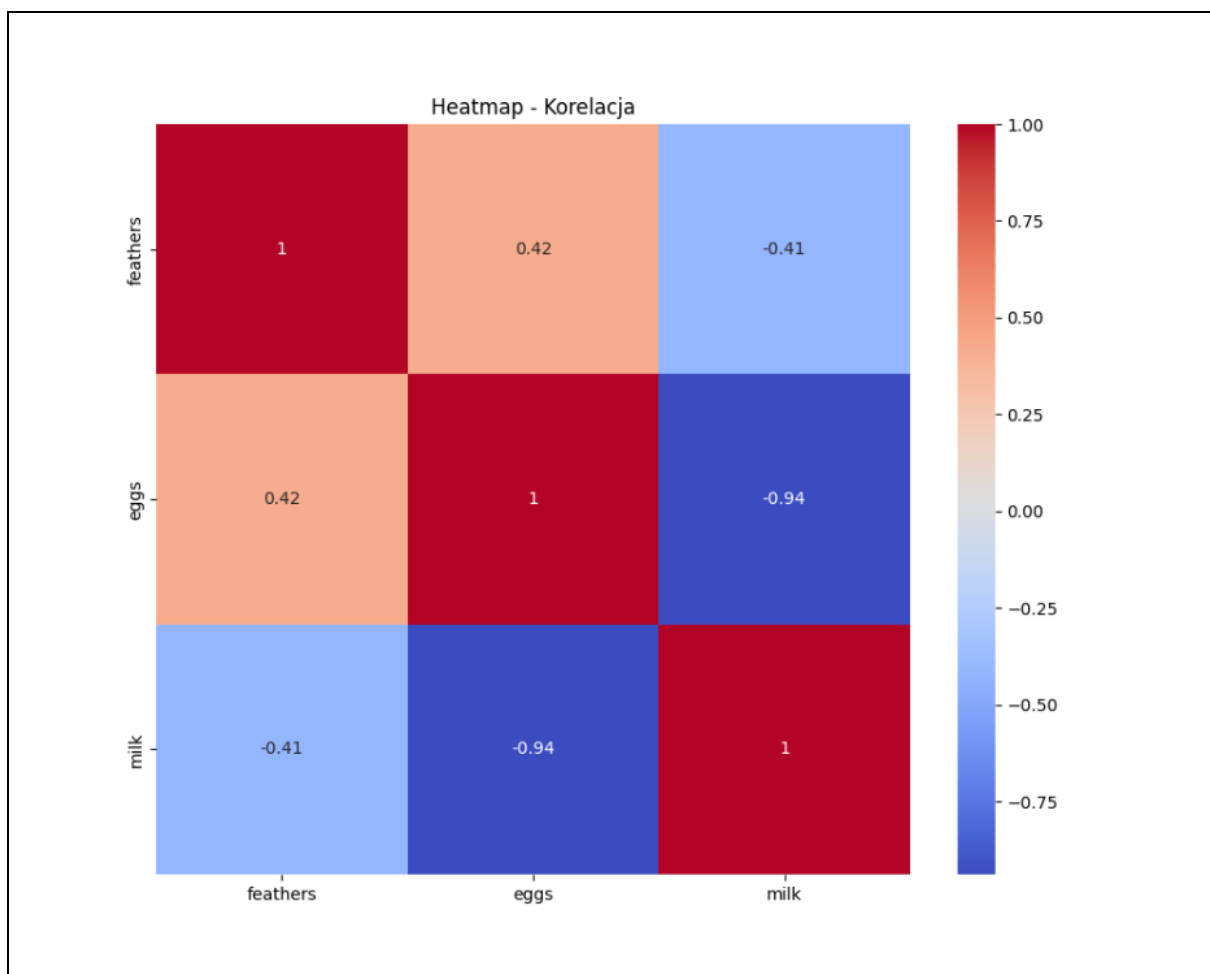
Każdy z tych wykresów pozwala nam wywnioskować inne informacje. Dla przykładu wykres zliczeń:



Pozwala nam wnioskować, że w naszej bazie danych dominują zwierzęta składające jaja, a jednocześnie około 80 z nich nie posiada piór.



Następnie, dzięki wykresowi dystrybucji, możemy precyzyjnie określić, że 49,2% zwierząt składających jaja dotyczy tych posiadających zarówno pióra, jak i karmiących mlekiem.



Ostatni z wykresów, heatmapa korelacji, potwierdza informację, że istnieje silna negatywna korelacja (-0,94) pomiędzy składaniem jaj a karmieniem mlekiem u zwierząt. Innymi słowy, zwierzęta składające jaja zazwyczaj nie karmią mlekiem, co stanowi istotny wniosek z analizy danych.

7.8. Klasyfikacja

Classification:

hair

Classificate

Jedną z funkcjonalności systemu jest klasyfikacja danych za pomocą algorytmów uczenia maszynowego. Aby rozpocząć klasyfikację zaznaczamy w tabeli kolumny danych na których algorytm będzie się uczył, następnie wybieramy kolumnę której zostanie stworzone przewidywanie wartości, a następnie porównane prawdziwymi.

PyNalise

File

	animal name	hair	feathers	eggs	milk	airbor
1	aardvark	1	0	0	1	0
2	antelope	1	0	0	1	0
3	bass	0	0	1	0	0
4	bear	1	0	0	1	0
5	boar	1	0	0	1	0
6	buffalo	1	0	0	1	0

feathers eggs milk Predicted hair

0 0.0 0.0 1.0 1.0
1 0.0 0.0 1.0 1.0
2 0.0 1.0 0.0 0.0
3 0.0 0.0 1.0 1.0
4 0.0 0.0 1.0 1.0
5 0.0 0.0 1.0 1.0
6 0.0 0.0 1.0 1.0
7 0.0 1.0 0.0 0.0
8 0.0 1.0 0.0 0.0
9 0.0 0.0 1.0 1.0
10 0.0 0.0 1.0 1.0
11 1.0 1.0 0.0 0.0

Classification:

hair

Classificate

Dokładność klasyfikacji: 0.9108910891089109

Po naciśnięciu przycisku Classificate zostanie wypisana tabela danych, w tym przypadku feather, eggs, milk oraz predicted hair które zostało sklasyfikowane za pomocą algorytmu drzewa decyzyjnego. Pod tą tabelą wypisana będzie również obliczona dokładność tej klasyfikacji. W tym przypadku jest to 91% dokładności co jest dobrym wynikiem biorąc pod uwagę, że danych przykładowych nie jest dużo.

7.9. Klastrowanie

Kolejną funkcjonalnością programu jest klastrowanie danych czyli grupowanie ich bez narzuconej etykiety.

PyNalise

File

	pw	pl	lw	ll	class
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5.0	3.6	1.4	0.2	Iris-setosa
6	5.4	3.9	1.7	0.4	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa

silhouette_avg:0.5496681486718054
Davies-Bouldin Index:0.6580527043577143

No info about Column

Classification:

pl

Classificate

Clusterization:

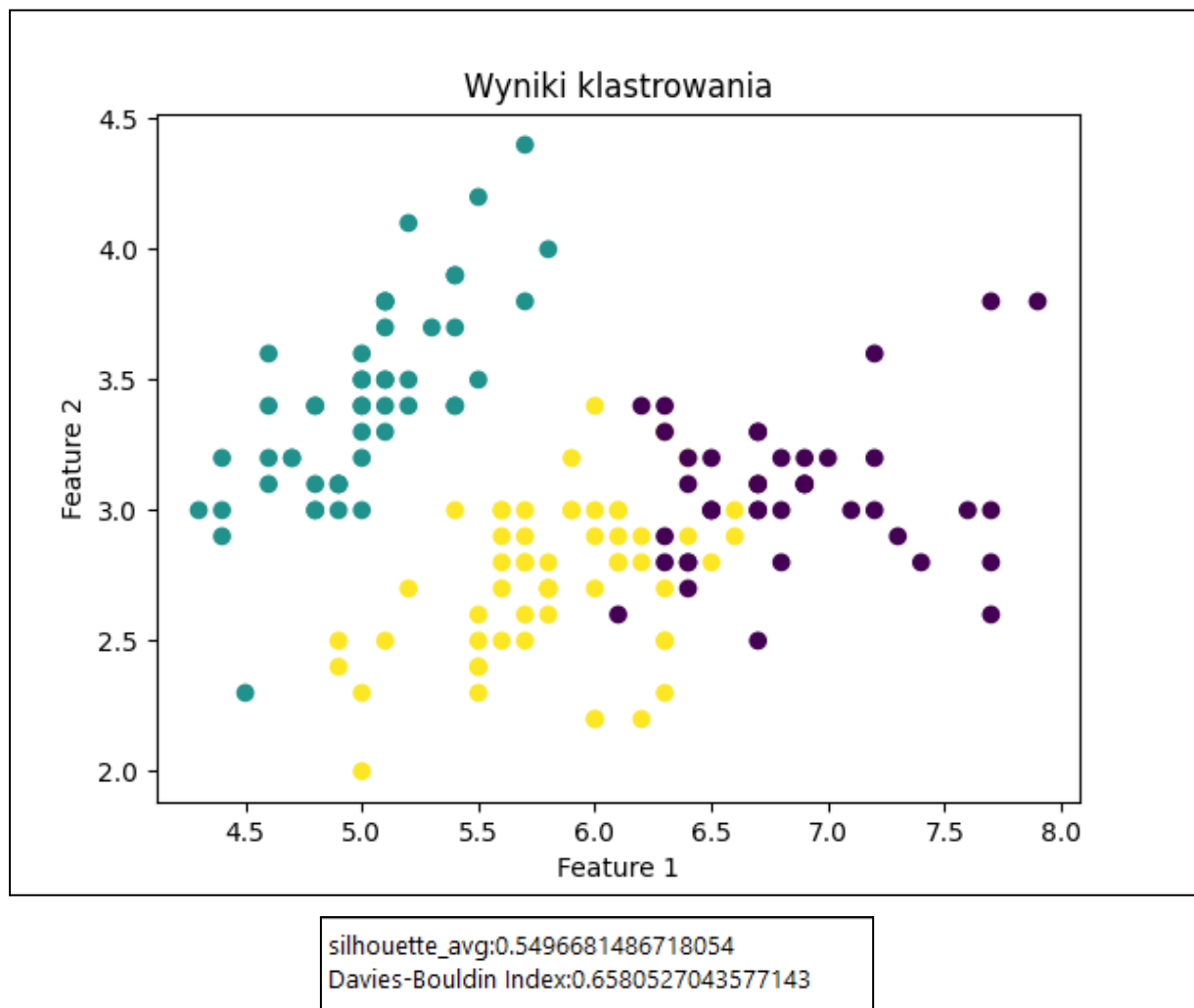
3

Cluster

Correlation:

Plots:

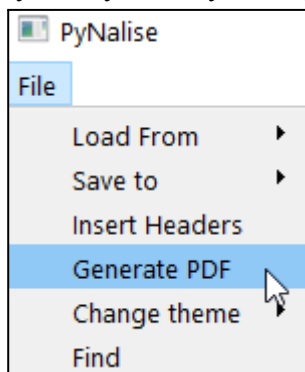
W dziale Klastracji w polu wejściowym wpisujemy liczbę klastrow. Następnie wciskamy przycisk Cluster aktywujący algorytm.



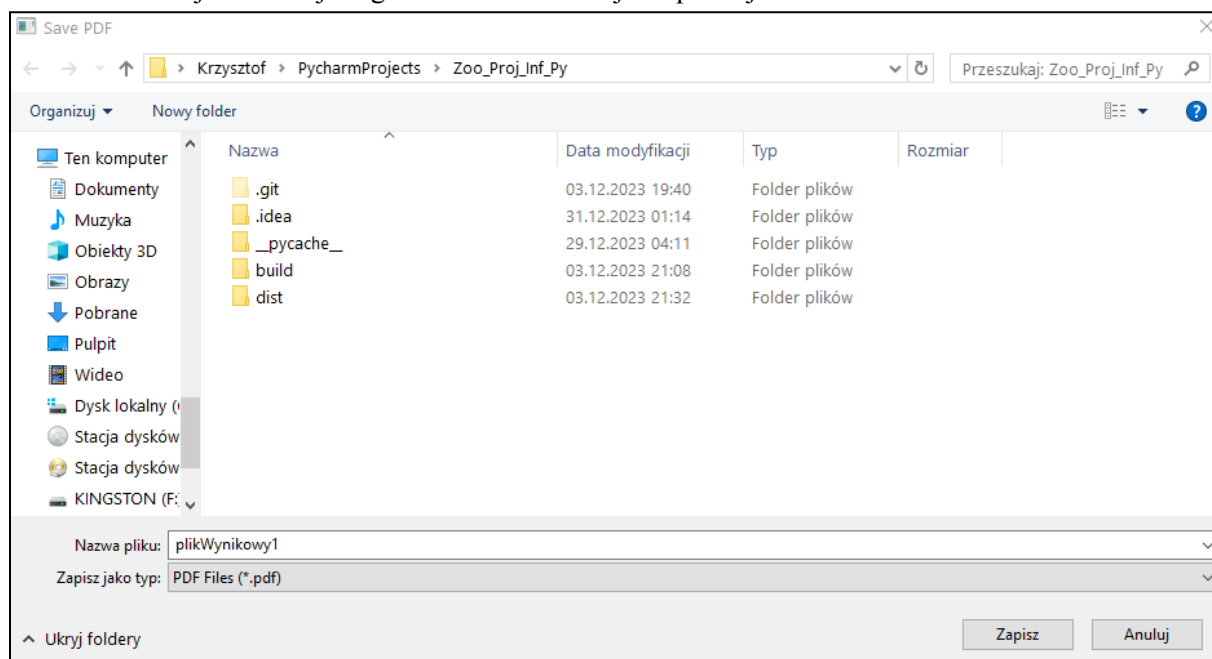
Wygenerowany zostaje wykres przedstawiający każdą grupę danych innym kolorem. W oknie wynikowym zostają wypisane wartości DBD oraz silhouette average które określają jakość klastracji na podstawie danych wejściowych.

7.10. Generowanie Pliku Wynikowego PDF

Aby wygenerować plik wynikowy PDF wybieramy tę opcję z menu File.



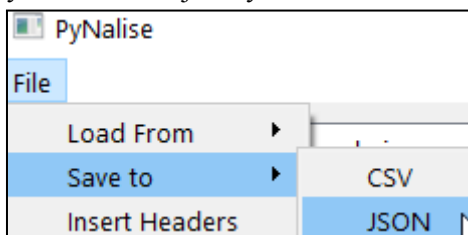
Następnie wybieramy miejsce na dysku w którym ma zostać zapisany plik. Domyślnie jest to miejsce gdzie zainstalowana jest aplikacja.



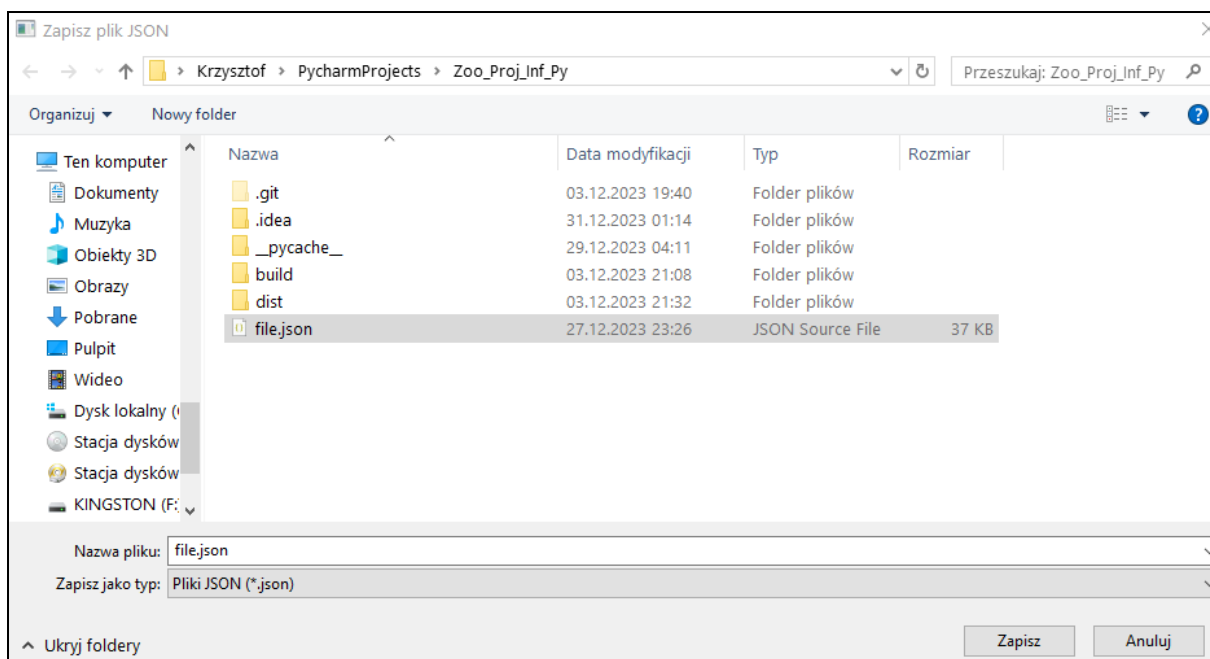
Po dokonaniu odpowiedniego wyboru zatwierdzamy nasz wybór i plik zostaje wygenerowany oraz zapisany na dysku.

7.11. Zapisywanie Danych

Możemy również zapisywać wcześniej wczytane dane do formatu CSV oraz JSON.



Wybieramy format pliku do jakiego chcemy zapisać dane z tabeli. Opcję klikamy w menu File->Save to->format



Następnie wybieramy miejsce docelowe zapisu naszego pliku na dysku i zatwierdzamy nasz wybór poprzez kliknięcie przycisku Zapisz.

8. Testowanie Błędów.

Testowanie błędów w aplikacji desktopowej jest kluczowym elementem procesu rozwoju oprogramowania, mającego na celu zapewnienie jakości i stabilności aplikacji.

8.1. Walidacja danych

Pierwszą funkcjonalnością, którą warto sprawdzić było logowanie oraz poprawność wprowadzanych danych. Istotną częścią walidacji danych jest sprawdzanie czy wprowadzane do bazy podczas rejestracji spełniają określone warunki i nie są szkodliwe, nie zawierają niechcianych znaków lub spełniają określone standardy jak adres email. Z testów tych nie wynikły żadne problemy.

8.2. Funkcje liczące dane liczbowe

Jednym z błędów na które natrafiłem podczas tworzenia aplikacji było przerywanie działania programu kiedy funkcje mające na celu obliczać coś na liczbach natrafiały na ciągi tekstowe. Rozwiązałem ten problem zamykając kod tych funkcji w specjalne wychwytywanie błędów, które sprawia że jeśli dane po naciśnięciu przycisku są nieprawidłowe to wypisane zostanie to na ekranie zamiast zakończyć działanie programu.

```
Error occurred: could not convert string to float: 'Iris-setosa'
```

Jest to jasna informacja dla użytkownika, że w ten sposób nie powinno się używać tej funkcjonalności.

8.3. Wczytywanie danych w formacie CSV

Format CSV cechuje się tym, że może posiadać nagłówki w pierwszej linii pliku lub może ich tam brakować. Aplikacja jest przygotowana na oba te scenariusze.

Użytkownik wybiera odpowiednią funkcję wiedząc jaki ma przygotowany plik dzięki czemu unikamy błędu w którym jako nagłówki byłby wczytywany pierwszy wiersz danych. Później możemy ręcznie wprowadzić również nagłówki już w systemie.

8.4. Wczytywanie nowych danych

Kolejnym wyzwaniem przy wczytywaniu danych jest odpowiednie zapisanie ich do DataFrame na których wykonywane są wszystkie obliczenia i która jest kluczową strukturą danych w aplikacji. Dodatkowo napisałem wątek który nieustannie wymienia dane między tabelą jako komponent interfejsu oraz DataFrame dzięki czemu dane które zedytujemy w tabeli odzwierciedlone są również pod powierzchnią frontálną systemu.

9. Podsumowanie

Projekt inżynierski Pynallise zakładał stworzenie kompleksowego systemu do analizy danych który miał zaspokoić potrzeby użytkowników w zakresie efektywnej analizy, interpretacji i prezentacji zgromadzonych informacji. Oto główne aspekty projektu, osiągnięcia oraz potencjalne obszary poprawy. Udało się stworzyć system z podziałem na role użytkowników, gdzie każdy użytkownik może wczytać dowolne dane z pliku CSV oraz JSON. Powiodła się implementacja algorytmów odpowiedzialnych za obliczenia statystyczne, za klasyfikację danych oraz klasteryzację. Udało się umożliwić użytkownikom edycję danych oraz zapisywanie wyników działania na programie. Udało się również wprowadzić konfigurację wizualną aplikacji.

Udało się omówić kluczowe znaczenie wykorzystania technologii Python w projekcie jak i jej innowacyjność oraz aktualność w dzisiejszym świecie. Ponadto całość projektu otrzymało godne miejsce reprezentacji na specjalnie dedykowanej do tego stronie internetowej również stworzonej w technologii Python. Założenia projektu zostały spełnione, jednak gdyby pracował nad nim zespół, który miałby na to więcej czasu niż ja mogłem na to poświęcić to rozwinięte mogłyby zostać następujące aspekty systemu:

- ilość, kompleksowość algorytmów odpowiedzialnych za klasyfikację oraz klastrację, wybrałem jedynie kilka z nich które wykorzystałem, jednak być może z niektórymi danymi lepiej poradziłyby sobie inne algorytmy. Biorąc pod uwagę cel projektu którym jest uniwersalność systemu jest to sfera możliwa do rozwinięcia
- połączenie bazy użytkowników systemu z bazą strony internetowej
- zwiększenie ilości konfiguracji systemu
- prezentacja wyników oraz tworzenie plików wynikowych mogłoby zostać rozwinięte
- ilość możliwych do wczytania formatów danych mogłoby zostać rozszerzony

10. Bibliografia

[pandas documentation — pandas 2.1.1 documentation \(pydata.org\)](#)

[Matplotlib documentation — Matplotlib 3.8.0 documentation](#)

[seaborn: statistical data visualization — seaborn 0.13.0 documentation \(pydata.org\)](#)

[Qt for Python](#)

[Encyklopedia Zarządzania \(mfiles.pl\)](#)

[PyQt6 · PyPI](#)

[PyQt6 Tutorial 2023, Create Python GUIs with Qt](#)

[scikit-learn: machine learning in Python — scikit-learn 1.3.2 documentation](#)

[scikit-learn · PyPI](#)

[seaborn: statistical data visualization — seaborn 0.13.0 documentation \(pydata.org\)](#)

[Python Tutorials – Real Python](#)