

Bazy Danych 2 - Projekt

Dokumentacja

Mateusz Cichostępski

18 czerwca 2023

Spis treści

1	Opis problemu	3
2	Opis funkcjonalności	3
3	Opis typów danych oraz metod udostępnionych przez API	8
4	Opis implementacji udostępnionego API przez bibliotekę	10
5	Przeprowadzone testy jednostkowe	10
5.1	Testy jednostkowe części użytkownika	10
6	Kod źródłowy	12
7	Podsumowanie i wnioski	13
7.1	Utworzenie bazy danych	13
7.2	Połączenie z bazą danych	13
7.3	Wnioski	13
8	Literatura	13

1 Opis problemu

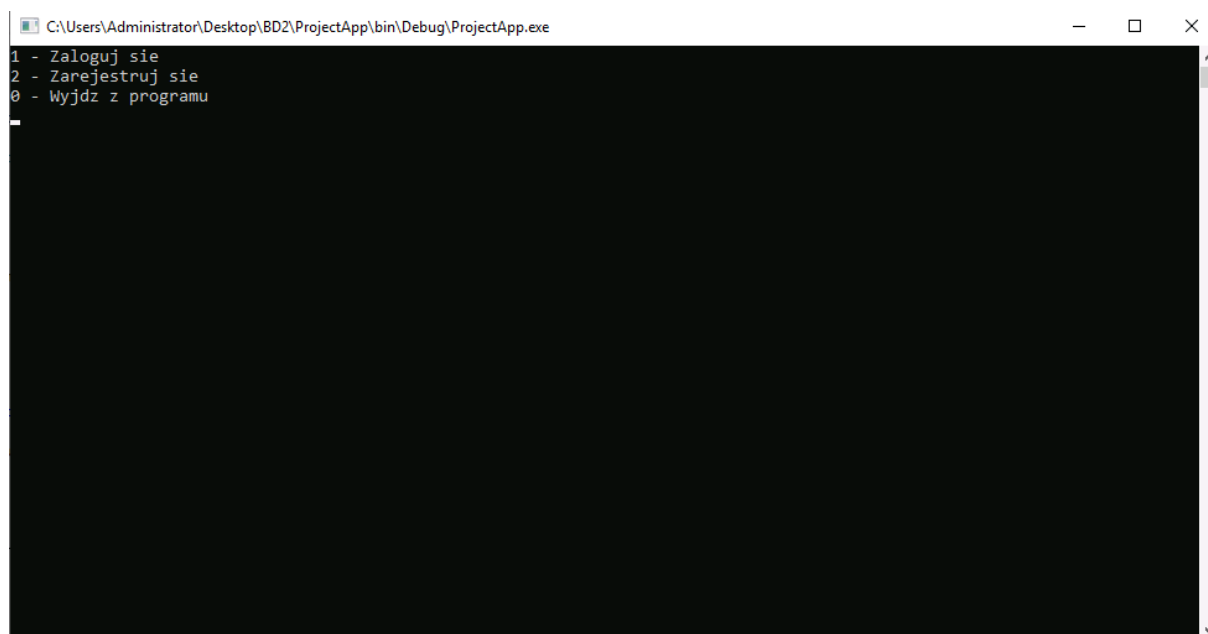
W ramach projektu opracowano bibliotekę z określonym interfejsem API do przetwarzania danych typu *CLOB*. *CLOB* (Character Large Object) to typ danych w bazach danych do przechowywania danych znakowych. Typ ten charakteryzuje się dużymi limitami na rozmiar (najczęściej do 2 GB). Niektóre systemy zarządzania bazami danych nie przechowują tekstu bezpośrednio w tabeli. Zamiast tego pole CLOB służy jako adres, który odwołuje się do lokalizacji danych.

MS SQL Server wspiera *CLOB* za pomocą typu *VARCHAR(MAX)*.

2 Opis funkcjonalności

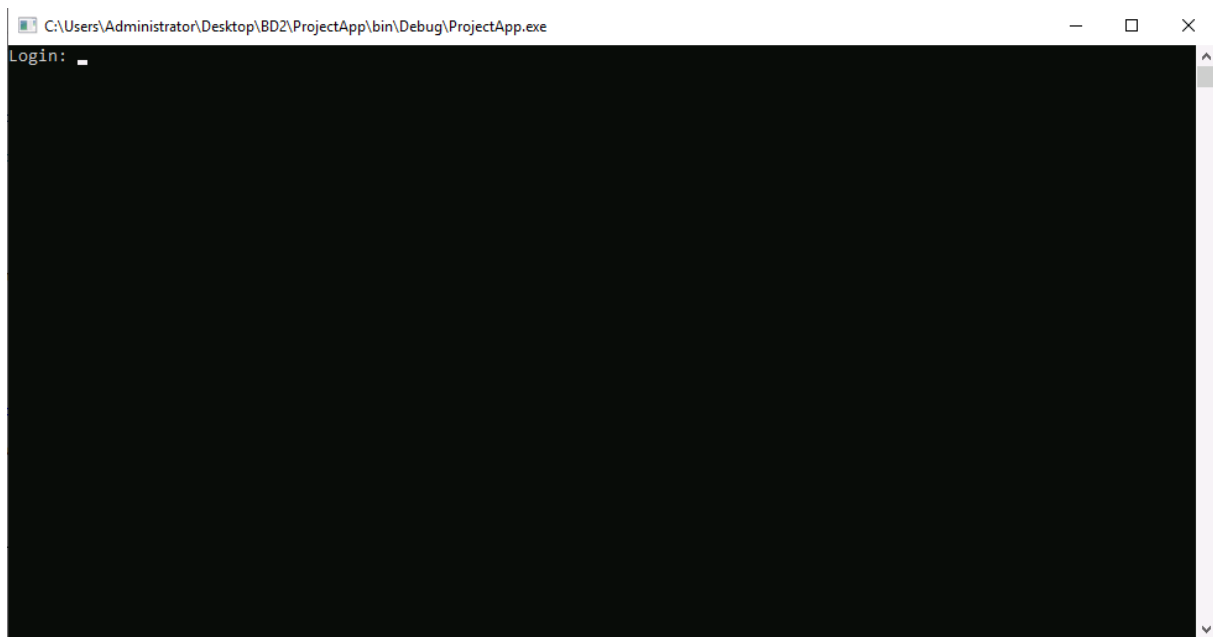
Stworzona aplikacja prezentująca funkcjonalności biblioteki API została napisana w języku C# jako aplikacja konsolowa. Użytkownik może wybierać odpowiednie akcje wyświetlone bezpośrednio w konsoli.

Po otwarciu aplikacji ukazuje się menu startowe z opcjami logowania bądź rejestracji. Po wpisaniu 1 oraz po potwierdzeniu enterem użytkownik przechodzi do strony z logowaniem (Rys 2). Po wpisaniu 2 użytkownik przechodzi do strony z rejestracją nowego użytkownika (Rys 3).



Rysunek 1: Menu główne

Na podstronie z logowaniem użytkownik podaje swoje login i hasło w celu weryfikacji tożsamości. Po poprawnym zalogowaniu użytkownik przechodzi na podstronę z akcjami dla zalogowanego użytkownika (Rys 4).



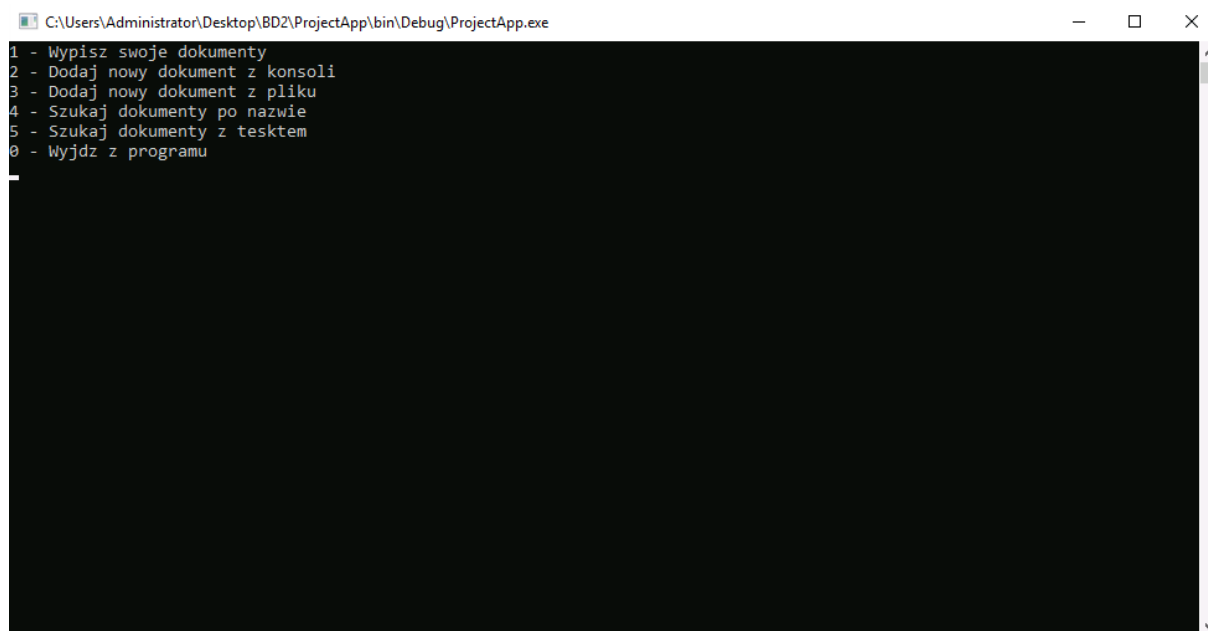
Rysunek 2: Podstrona z logowaniem

Na podstronie z rejestracją użytkownik może wpisać swój login i hasło. Jeżeli rejestracja przebiegnie pomyślnie, użytkownik ma możliwość zalogowania się.



Rysunek 3: Podstrona z rejestracją

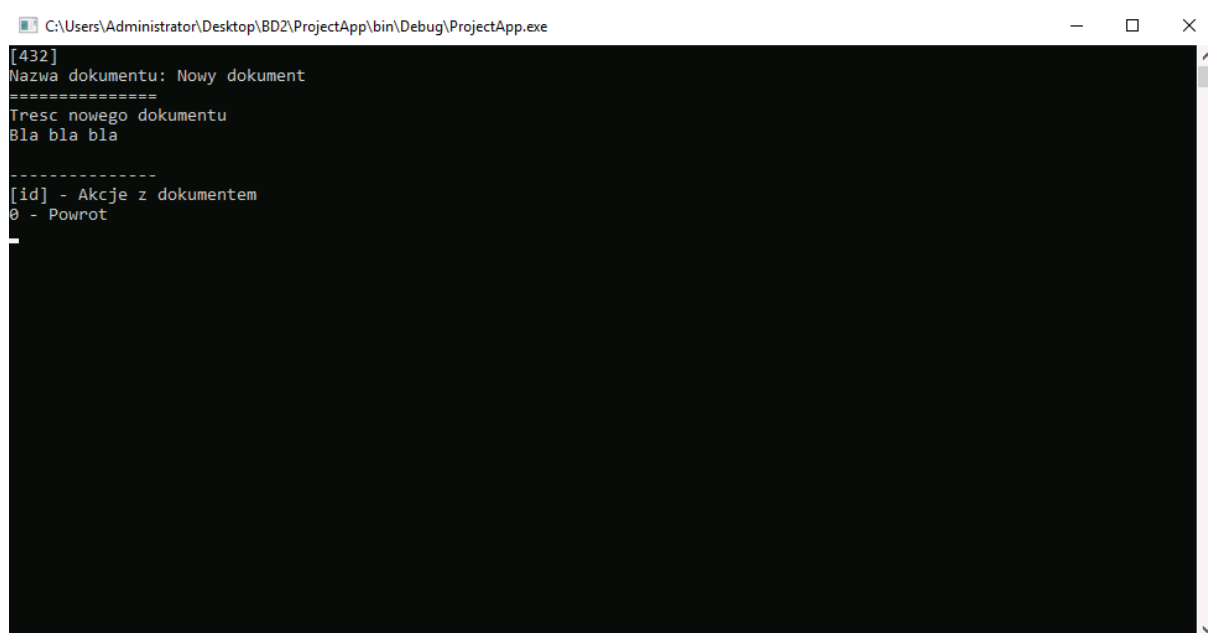
Zalogowany użytkownik może: wyświetlić wszystkie swoje dokumenty wybierając 1 (Rys 5), dodać nowy dokument wpisując jego zawartość w konsoli wybierając 2 (Rys 8), dodać nowy dokument z pliku tekstowego wybierając 3 (Rys 9), szukać dokumenty po nazwie dokumentu wybierając 4 (Rys 10), szukać dokumentów po tekście wybierając 5 (Rys 11).



```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
1 - Wypisz swoje dokumenty
2 - Dodaj nowy dokument z konsoli
3 - Dodaj nowy dokument z pliku
4 - Szukaj dokumenty po nazwie
5 - Szukaj dokumenty z tekstem
0 - Wyjdz z programu
```

Rysunek 4: Podstrona z akcjami zalogowanego użytkownika

Użytkownik ma wgląd do swoich dokumentów, a także wpisując odpowiedni *id* ukażą się użytkownikowi akcje związane z danym dokumentem (Rys 6).

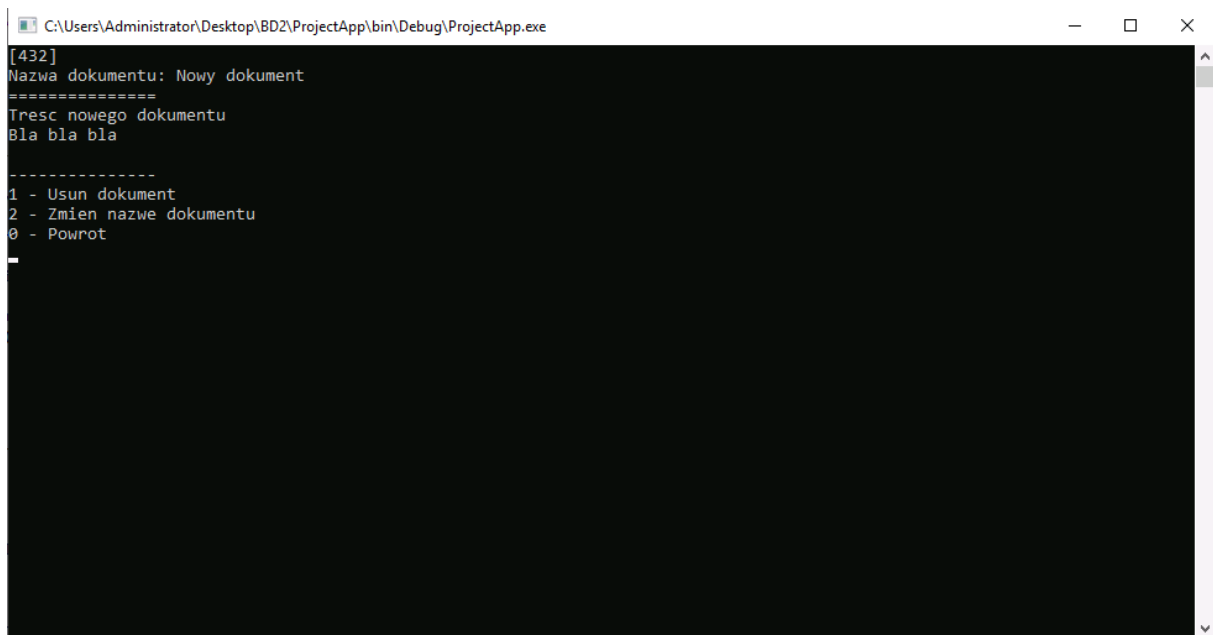


```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
[432]
Nazwa dokumentu: Nowy dokument
=====
Tresc nowego dokumentu
Bla bla bla

-----
[id] - Akcje z dokumentem
0 - Powrot
```

Rysunek 5: Podstrona z wszystkimi dokumentami użytkownika

Użytkownik ma wgląd do danego dokumentu oraz posiada akcje z nim związane. Może usunąć dokument wybierając 1 lub zmienić nazwę dokumentu wybierając 2 (Rys 7).

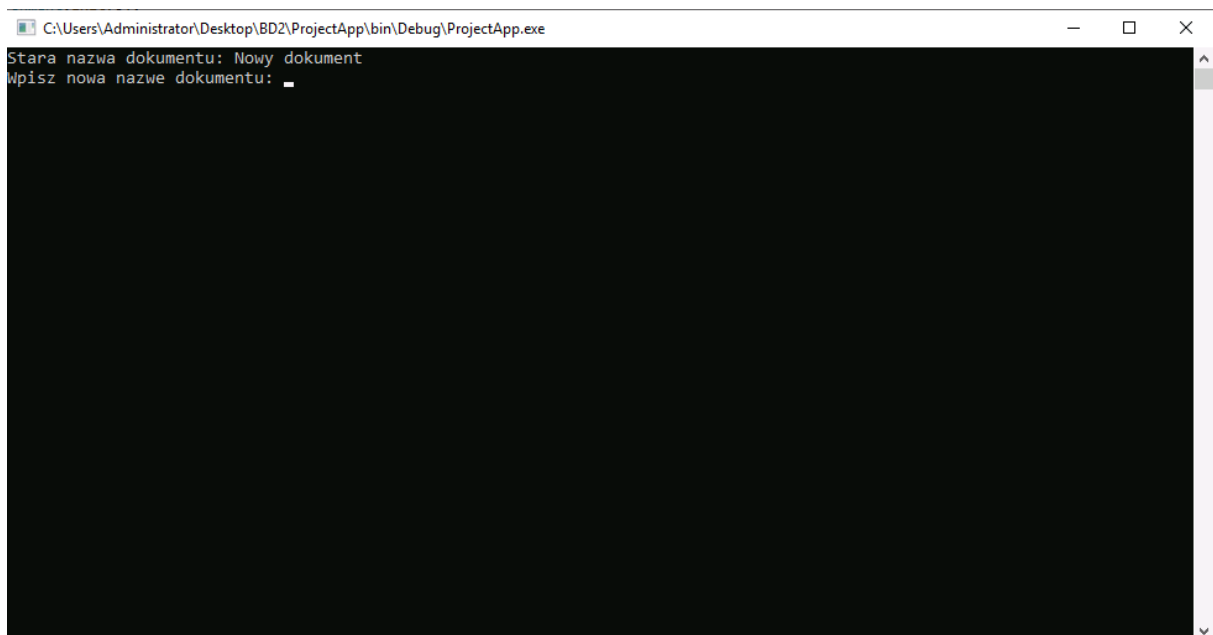


```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
[432]
Nazwa dokumentu: Nowy dokument
=====
Tresc nowego dokumentu
Bla bla bla

-----
1 - Usun dokument
2 - Zmien nazwe dokumentu
0 - Powrot
_
```

Rysunek 6: Podstrona z akcjami z dokumentem

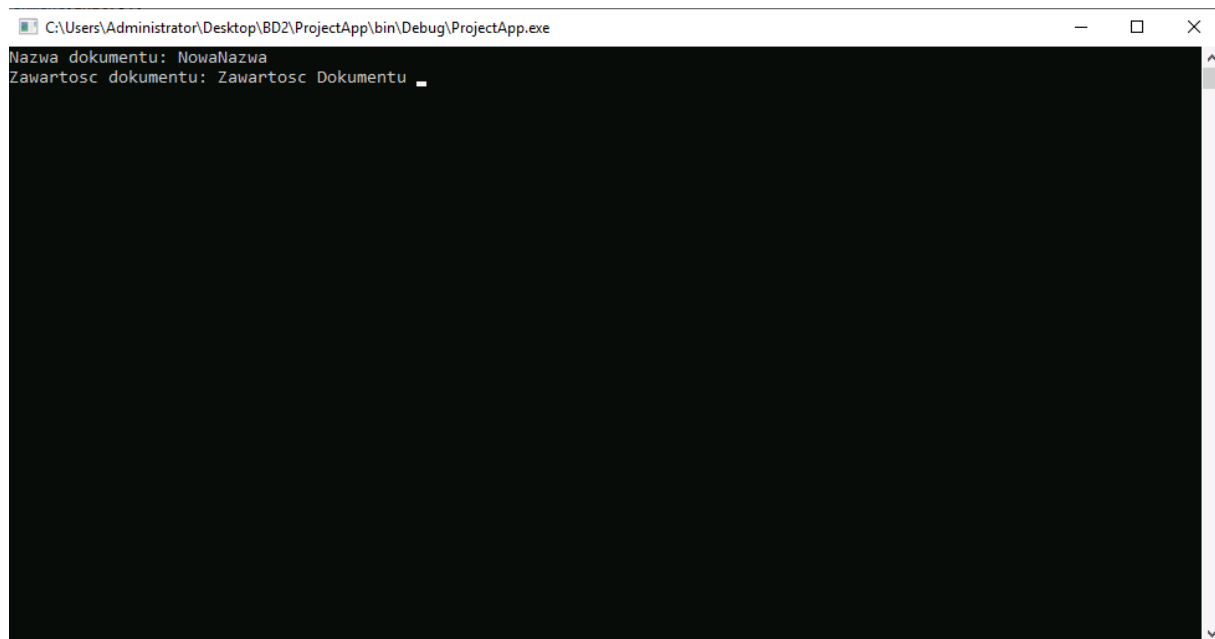
Użytkownik może ustawić nową nazwę dokumentu. Na ekranie wyświetla mu się także poprzednia nazwa.



```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
Stara nazwa dokumentu: Nowy dokument
Wpisz nowa nazwe dokumentu: _
```

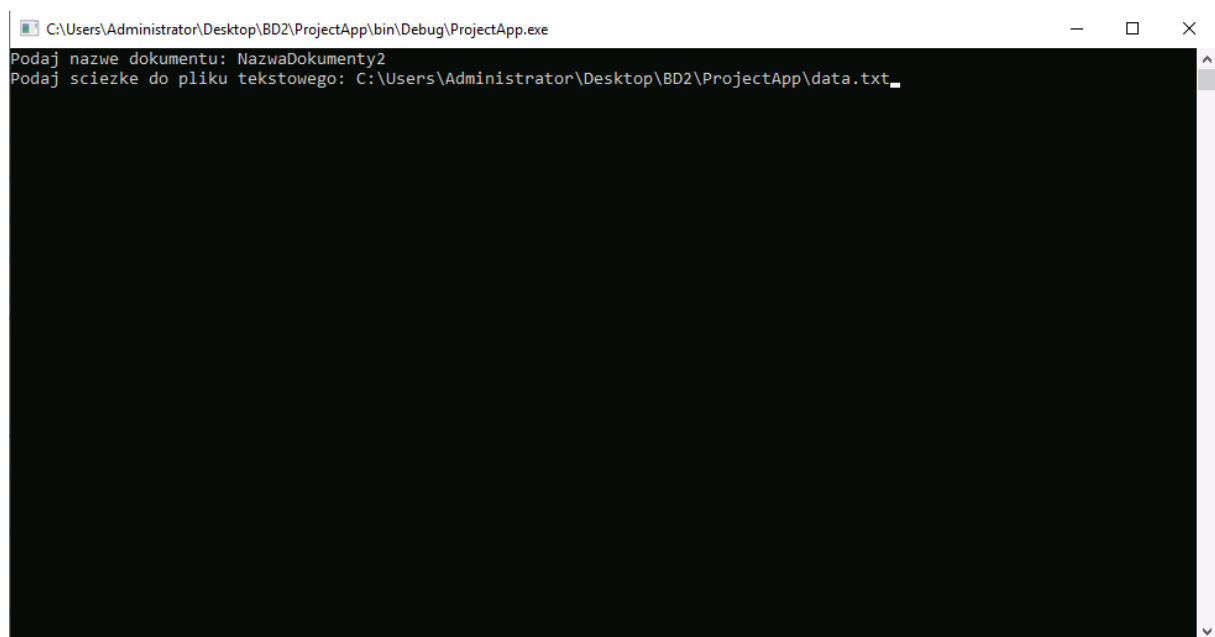
Rysunek 7: Podstrona z zmianą nazwy dokumentu

Użytkownik wpisuje nazwę dokumentu, a także jego zawartość. Aby zapisać należy nacisnąć dwa razy enter.



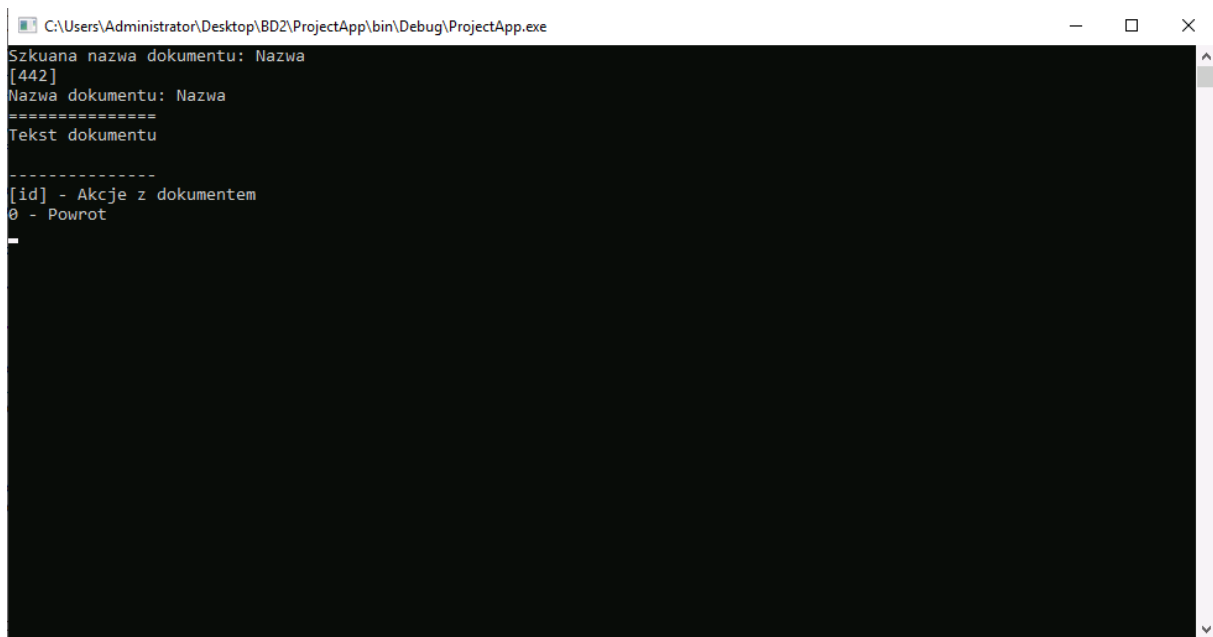
Rysunek 8: Podstrona z tworzeniem dokumentu

Użytkownik wpisuje nazwę dokumentu, a następnie ścieżkę do pliku.



Rysunek 9: Podstrona z tworzeniem dokumentu za ścieżki do pliku tekstowego

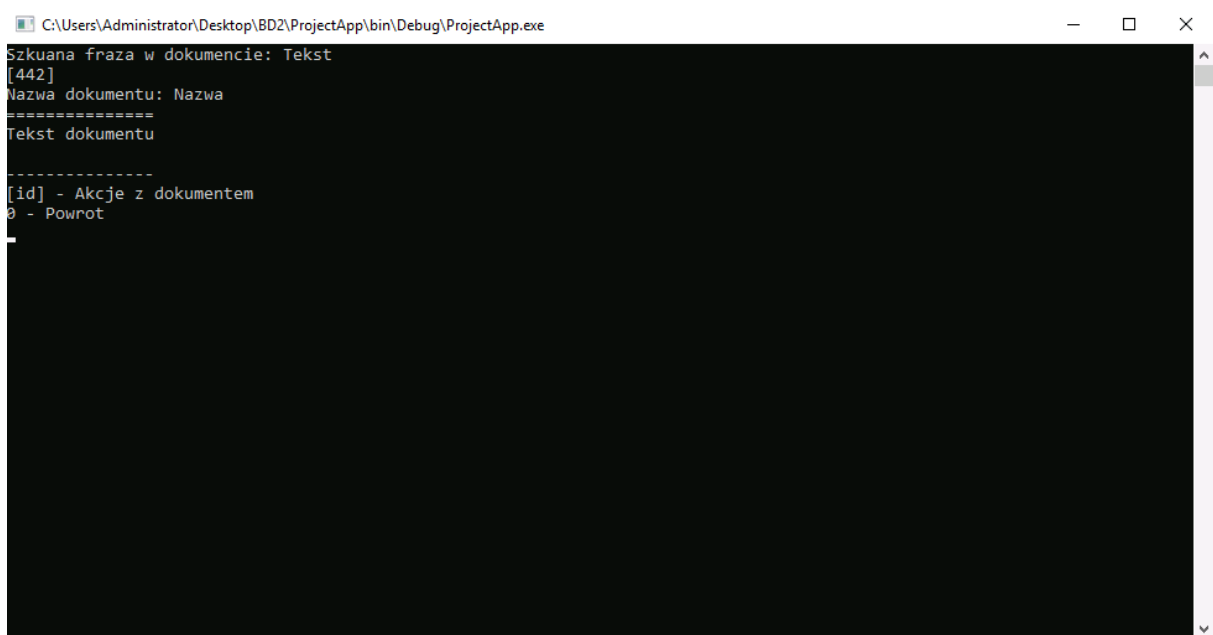
Użytkownik wpisuje frazę, którą zawiera nazwa jego dokumentu.



```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
Szukana nazwa dokumentu: Nazwa
[442]
Nazwa dokumentu: Nazwa
=====
Tekst dokumentu
-----
[id] - Akcje z dokumentem
0 - Powrot
```

Rysunek 10: Podstrona z szukaniem dokumentu z podaną nazwą

Użytkownik wpisuje dowolną frazę, która ma zostać odnaleziona w jego dokumentach. Program wyświetli te dokumenty.



```
C:\Users\Administrator\Desktop\BD2\ProjectApp\bin\Debug\ProjectApp.exe
Szukana fraza w dokumencie: Tekst
[442]
Nazwa dokumentu: Nazwa
=====
Tekst dokumentu
-----
[id] - Akcje z dokumentem
0 - Powrot
```

Rysunek 11: Podstrona z wyszukiwaniem podanej frazy w dokumentach

3 Opis typów danych oraz metod udostępnionych przez API

Biblioteka API udostępnia dane metody związane z przetwarzaniem użytkownika:

- ***ProjectFunctions***(*string connString*) - konstruktor tworzący instancję całej biblioteki, jedynym argumentem jest connection string do bazy danych użytkownika ("Data Source=WINSERV01;...")
- ***int createUser***(*string login*, *string password*) - metoda tworząca nowego obiekt użytkownika w bazie danych, jako argumenty podaje się login oraz hasło. Rzuca wyjątek jeżeli podany login już istnieje. Zwraca 1 jeżeli wszystko przebiegło pomyślnie.

- *int* **updatePassword**(*string oldPassword, string newPassword*) - metoda zmieniająca hasło użytkownika na nowe. Rzuca wyjątki jeżeli użytkownik nie jest zalogowany lub gdy podane stare hasło jest błędne. Zwraca *1* jeżeli wszystko przebiegnie pomyślnie.
- *int* **deleteUser**() - metoda do usunięcia użytkownika z bazy danych. Zwraca *1* jeżeli wszystko przeszło pomyślnie.
- *bool* **loginUser**(*string login, string password*) - metoda do zalogowania użytkownika. Argumentami są login i hasło. Rzuca wyjątki jeżeli podany login nie będzie istniał lub gdy hasło nie będzie się zgadzało. Zwraca *true* wszystko przebiegnie pomyślnie.
- *void* **logoutUser**() - metoda wylogowująca użytkownika.

Biblioteka API udostępnia dane metody związane z przetwarzaniem typu *CLOB*:

- *int* **createClobObjectFromString**(*string document, string name*) - metoda tworząca obiekt dokumentu z bazie danych przypisany do użytkownika. Przyjmuje treść dokumentu typu *CLOB* oraz nazwę dokumentu. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany. Zwraca *1* jeżeli wszystko przebiegnie pomyślnie.
- *int* **createClobObjectFromFile**(*string filename, string name*) - metoda tworząca obiekt dokumentu w bazie danych przypisany do użytkownika. Jako argument przyjmuje ścieżkę do pliku tekstowego oraz nazwę dokumentu. Rzuca wyjątkami jeżeli użytkownik nie jest zalogowany, podana ścieżka nie istnieje lub użytkownik posiada niewystarczającą ilość pamięci RAM do przetworzenia typu *CLOB*. Zwraca *1* jeżeli proces przebiegnie pomyślnie.
- *List*<*PUserDocuments*> **searchDocumentsByText**(*string textToSearch*) - metoda do wyszukiwania dokumentów danego użytkownika z podaną frazą tekstową. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany. Zwraca listę z dokumentami.
- *int*[] **searchDocumentIdsByText**(*string textToSearch*) - metoda zwracająca tablice z id dokumentów danego użytkownika zawierającymi podaną frazę tekstową. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany.
- *string*[] **searchDocumentNamesByText**(*string textToSearch*) - metoda zwracająca tablice z nazwami dokumentów danego użytkownika zawierającymi podaną frazę tekstową. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany.
- *List*<*PUserDocuments*> **getUserDocuments**() - metoda zwracająca wszystkie dokumenty użytkownika. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany.
- *int* **updateNameOfDocument**(*int documentId, string newName*) - metoda zmieniająca nazwę danego dokumentu użytkownika. Jako argumenty przyjmuje id dokumentu oraz nową nazwę. Rzuca wyjątki jeżeli użytkownik nie jest zalogowany, nowa nazwa nie spełnia wymagań lub dokument nie istnieje. Zwraca *1* jeżeli proces przebiegnie pomyślnie.
- *int* **deleteDocumentWithId**(*int documentId*) - metoda usuwająca dany dokument użytkownika. Jako argument przyjmuje id dokumentu. Rzuca wyjątkami jeżeli użytkownik nie jest zalogowany lub podany dokument nie istnieje. Zwraca *1* jeżeli proces przebiegnie pomyślnie.
- *int*[] **searchDocumentIdsByName**(*string documentName*) - metoda zwracająca tablicę id dokumentów użytkownika o podanej nazwie. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany.
- *List*<*PUserDocuments*> **searchDocumentByName**(*string documentName*) - metoda zwracająca listę z dokumentami użytkownika zawierającymi podaną nazwę. Rzuca wyjątek jeżeli użytkownik nie jest zalogowany.

Typy zwracane przez metody biblioteki API:

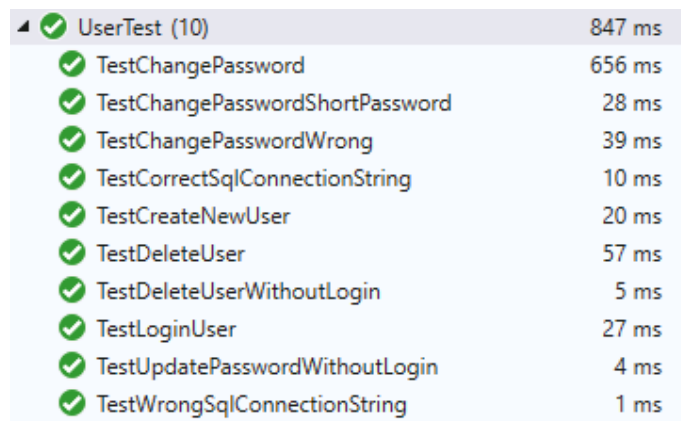
- Klasa **PUserDocuments**
 - *int id* - id dokumentu
 - *string name* - nazwa dokumentu
 - *string document* - zawartość dokumentu

4 Opis implementacji udostępnionego API przez bibliotekę

5 Przeprowadzone testy jednostkowe

5.1 Testy jednostkowe części użytkownika

Napisano 10 testów jednostkowych sprawdzających funkcje udostępnione przez bibliotekę związane z zarządzaniem użytkownika.



▲ ✓ UserTest (10)	847 ms
✓ TestChangePassword	656 ms
✓ TestChangePasswordShortPassword	28 ms
✓ TestChangePasswordWrong	39 ms
✓ TestCorrectSqlConnectionString	10 ms
✓ TestCreateNewUser	20 ms
✓ TestDeleteUser	57 ms
✓ TestDeleteUserWithoutLogin	5 ms
✓ TestLoginUser	27 ms
✓ TestUpdatePasswordWithoutLogin	4 ms
✓ TestWrongSqlConnectionString	1 ms

Rysunek 12: Przeprowadzone testy jednostkowe związane z zarządzaniem użytkownikiem

- ***TestChangePassword*** - test sprawdzający zmianę hasła użytkownika. Test tworzy nowego użytkownika, loguje się na niego, próbuje zmienić hasło. Sprawdza poprawność zmiany hasła poprzez ponowne zalogowanie.
- ***TestChangePasswordShortPassword*** - test sprawdzający zmianę hasła użytkownika jeżeli poda się zbyt krótkie hasło, w tym przypadku pusty string. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestChangePasswordWrong*** - test sprawdzający zmianę hasła użytkownika jeżeli poda błędne obecne hasło. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestCorrectSqlConnectionString*** - test sprawdzający poprawność wprowadzonego connection string przez aplikację.
- ***TestCreateNewUser*** - test sprawdzający poprawność tworzenia nowego użytkownika. Tworzy nowego użytkownika z losowo wygenerowanymi loginem i hasłem, a następnie próbuje się nimi zalogować.
- ***TestDeleteUser*** - test sprawdzający poprawność usunięcia użytkownika. Test tworzy nowego użytkownika, usuwa go oraz sprawdza możliwość ponownego zalogowania się. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestDeleteUserWithoutLogin*** - test sprawdzający usunięcie użytkownika jeżeli ten nie jest zalogowany. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestLoginUser*** - test sprawdzający poprawność logowania się. Tworzy nowego użytkownika z losowo wygenerowanymi loginem i hasłem, sprawdza możliwość zalogowania się.
- ***TestUpdatePasswordWithoutLogin*** - test sprawdzający brak możliwości zmiany hasła bez uprzedniego zalogowania się użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestWrongSqlConnectionString*** - test sprawdzający błędnie podany connection string do aplikacji. Sprawdza poprawność wyrzucenia wyjątku.

Napisano 24 testy jednostkowe dla części związanej z przetwarzaniem typu *CLOB* udostępnionej przez API.

▲ ✓ ClobTest (24)	6,5 sec
✓ TestAddClobFromFile	4,1 sec
✓ TestAddClobFromFileThatNotExists	48 ms
✓ TestAddClobFromFileWithoutLoggedIn	12 ms
✓ TestAddClobFromString	69 ms
✓ TestAddClobFromStringWithoutLoggedIn	11 ms
✓ TestChangeDocumentNameWithoutLog...	10 ms
✓ TestChangeDocumentNameWithWrong...	102 ms
✓ TestChangeNameOfDocument	246 ms
✓ TestDeleteDocument	278 ms
✓ TestDeleteDocumentWithoutLoggedIn	29 ms
✓ TestDeleteDocumentWithWrongId	116 ms
✓ TestGetUserDocuments	156 ms
✓ TestGetUserDocumentsWithoutLoggedIn	13 ms
✓ TestSearchDocumentByNameWithoutLo...	10 ms
✓ TestSearchDocumentByText	216 ms
✓ TestSearchDocumentByTextWithoutLogg...	12 ms
✓ TestSearchDocumentIdByText	223 ms
✓ TestSearchDocumentIdByTextWithoutLo...	9 ms
✓ TestSearchDocumentIdsByName	316 ms
✓ TestSearchDocumentIdsByNameWithout...	30 ms
✓ TestSearchDocumentNamesByText	157 ms
✓ TestSearchDocumentNamesByTextWitho...	8 ms
✓ TestSearchDocumentsByName	188 ms
✓ TestWrongNewNameOfDocument	125 ms

Rysunek 13: Przeprowadzone testy jednostkowe związane z przetwarzaniem typu *CLOB*

- ***TestAddClobFromFile*** - test sprawdzający poprawność tworzenia dokumentu z pliku tekstowego. Test loguje przykładowego użytkownika i dodaje do niego nowy dokument podając ścieżkę do istniejącego pliku tekstowego.
- ***TestAddClobFromFileThatNotExists*** - test sprawdzający podanie błędnej ścieżki do pliku. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestAddClobFromFileWithoutLoggedIn*** - test sprawdzający próbę stworzenia dokumentu bez uprzedniego zalogowania się użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestAddClobFromString*** - test sprawdzający poprawność tworzenia dokumentu poprzez przekazanie bezpośrednio wartości do API.
- ***TestAddClobFromStringWithoutLoggedIn*** - test sprawdzający próbę stworzenia dokumentu poprzez przekazanie bezpośrednio wartości bez uprzedniego zalogowania się użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestChangeDocumentNameWithoutLoggedIn*** - test sprawdzający próbę zmiany nazwy dokumentu bez uprzedniego zalogowania się użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestChangeDocumentNameWithWrongDocumentId*** - test sprawdzający próbę zmiany nazwy dokumentu z błędnie podanym id dokumentu, który nie należy do użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestChangeNameOfDocument*** - test sprawdzający poprawność zmiany nazwy dokumentu. Test tworzy nowego użytkownika, przykładowy 1 dokument, zmienia nazwę tego dokumentu a następnie sprawdza poprawność nowej nazwy pobierając na nowo dokumenty użytkownika.

- ***TestDeleteDocument*** - test sprawdzający poprawność usunięcia dokumentu użytkownika. Test tworzy nowego użytkownika, losowe dokumenty, usuwa pierwszy z nich a następnie sprawdza ilość dokumentów zwracaną przez API.
- ***TestDeleteDocumentWithoutLoggedIn*** - test sprawdzający próbę usunięcia dokumentu bez wcześniejszego zalogowania użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestDeleteDocumentWithWrongId*** - test sprawdzający próbę usunięcia dokumentu z błędnym podanym id dokumentu. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestGetUserDocuments*** - test sprawdzający poprawność pobrania dokumentów użytkownika. Test tworzy nowego użytkownika, dodaje do niego określoną liczbę dokumentów a następnie sprawdza tożsamość zwróconych dokumentów (nazwę oraz zawartość).
- ***TestGetUserDocumentsWithoutLoggedIn*** - test sprawdzający próbę pobrania dokumentów niezalogowanego użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentByNameWithoutLoggedIn*** - test sprawdzający próbę wyszukania dokumentu bez uprzedniego zalogowania użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentByText*** - test sprawdzający poprawność wyszukiwania tekstu w dokumentach użytkownika. Test tworzy nowego użytkownika, przykładowe dokumenty a następnie sprawdza ilość zwróconych dokumentów przez API.
- ***TestSearchDocumentByTextWithoutLoggedIn*** - test sprawdzający próbę wyszukiwania tekstu w dokumentach bez uprzedniego zalogowania użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentIdByText*** - test sprawdzający ilość zwróconych elementów indeksów dokumentów z API poprzez szukanie tekstu w dokumentach.
- ***TestSearchDocumentIdByTextWithoutLoggedIn*** - test sprawdzający próbę wyszukania indeksów dokumentów z podanym tekstem bez uprzedniego zalogowania się. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentIdsByName*** - test sprawdzający szukanie indeksów dokumentów po nazwie. Test tworzy nowego użytkownika, dodaje kilka przykładowych dokumentów a następnie sprawdza ilość indeksów dokumentów zwróconą przez API.
- ***TestSearchDocumentIdsByNameWithoutLoggedIn*** - test sprawdzający próbę wyszukania dokumentów po nazwach bez uprzedniego zalogowania się użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentNamesByText*** - test sprawdzający zwracane nazwy dokumentów zawierające podany tekst. Test tworzy nowego użytkownika, dodaje do niego przykładowe dokumenty a następnie sprawdza tożsamość nazw zwróconych dokumentów przez API.
- ***TestSearchDocumentNamesByTextWithoutLoggedIn*** - test sprawdzający próbę wyszukania nazw dokumentów zawierających dany tekst bez uprzedniego zalogowania użytkownika. Sprawdza poprawność wyrzucenia wyjątku.
- ***TestSearchDocumentsByName*** - test sprawdzający szukanie dokumentów po ich nazwie. Test tworzy nowego użytkownika, dodaje kilka przykładowych dokumentów a następnie sprawdza ilość dokumentów zwróconą przez API.
- ***TestWrongNewNameOfDocument*** - test sprawdzający próbę zmiany nazwy dokumentu na niepoprawną, w tym przypadku pusty string. Sprawdza poprawność wyrzucenia wyjątku.

6 Kod źródłowy

Cały kod projektu można znaleźć pod tym linkiem: <https://github.com/Matiixx/BD2-Project>.
Znajdują się tam pliki pozwalające na odtworzenie projektu.

- W folderze **Project** znajduje się kod implementacji biblioteki API.

- W folderze **ProjectApp** znajduje się kod implementacji prostej aplikacji konsolowej do zaprezentowania funkcjonalności biblioteki.
- W folderze **ProjectTest** znajdują się pliki źródłowe testów biblioteki API.
- W folderze **SQL** znajdują się skrypty *SQL* pozwalające na stworzenie odpowiedniej struktury bazy danych.

7 Podsumowanie i wnioski

7.1 Utworzenie bazy danych

Aby utworzyć odpowiednią architekturę bazy danych i odpowiednie tabele należy wywołać skrypty z folderu *SQL* w odpowiedniej kolejności. Najpierw *SQL1*, a następnie *SQL2*.

7.2 Połączenie z bazą danych

W *ProjectApp* należy ustawić odpowiedni connection string. W pliku *MainProgram.cs* w linii 18.

7.3 Wnioski

Biblioteka jak i aplikacja konsolowa zostały napisane w całości w języku *C#*. Opracowane API umożliwia przetwarzanie danych typu *CLOB*. Typ używany do przechowywania dużych ilości tekstowych informacji. W ramach projektu wykorzystano również technologię wyszukiwania pełnotekstowego FTS (Full-Text Search), która umożliwia efektywne wyszukiwanie tekstów w bazie danych.

W trakcie implementacji API, zaimplementowano funkcje umożliwiające tworzenie i usuwanie użytkowników, a także tworzenie, wyszukiwanie i usuwanie dokumentów z zawartością *CLOB*.

Niestety materiały i maszyny komputerowe udostępnione przez uczelnię nie pozwoliły w pełni wykorzystać potencjału typu danych *CLOB*.

8 Literatura

- <https://docs.oracle.com/cloud/help/pl/analytics-cloud/ACSMD/GUID-6C8C333C-FF56-4376-9914-C298DF0D01htm#BIPDM207>
- <http://herongyang.com/JDBC/SQL-Server-CLOB-Large-Object.html>
- <https://learn.microsoft.com/en-us/sql/connect/jdbc/using-advanced-data-types?view=sql-server-ver16>
- <https://docs.oracle.com/javadb/10.10.1.2/ref/rrefclob.html>