



**Silesian University
of Technology**

Tabelkomistrz

Dokumentacja projektu

Mateusz Białecki, Alicja Chmielewska, Paweł Czyż,
Kamila Gendasz

20 stycznia 2022

Spis treści

1	Wstęp, założenia projektu	2
2	Praca nad projektem	3
2.1	Dobór narzędzi	3
2.2	Wyszczególnienie zadań	4
3	Efekt końcowy	4
3.1	Gdzie dostępny jest wynik końcowy	4
3.2	Instrukcja obsługi programu	5
3.2.1	Zmiana rozmiaru tabeli	5
3.2.2	Stylizacja tekstu w komórkach	6
3.2.3	Scalanie i rozdzielanie komórek	7
3.2.4	Szablony	8
3.2.5	Eksport	9
4	Podsumowanie	10
4.1	Dalszy rozwój	10
5	Bibliografia	11

1 Wstęp, założenia projektu

Tworzenie tabel w systemie LaTeX przez niektórych może być uznane za nieco nieintuicyjne, trudne w szybkim, sprawnym tworzeniu, żmudne przy formatowaniu. Gdy tabel takich nie tworzy się na co dzień łatwo zapomnieć skomplikowanej składni i zasad scalania poszczególnych komórek.

Nasz projekt „Tabelkomistrz” zakłada stworzenie programu pomocnego przy tworzeniu tabel w LaTeX. Użytkownik, za pomocą przyjaznego środowiska graficznego, ma być w stanie w łatwy, szybki i przyjemny sposób stworzyć tabelę w trybie WYSIWYG, której LaTeX-owy kod zostanie mu zwrócony.

Program posiada zaimplementowane funkcjonalności takie jak:

- Tworzenie tabeli w trybie WYSIWYG
- Scalanie i rozdzielanie komórek
- Eksportowanie do kodu LaTeX
- Stylizowanie treści komórek (pogrubienie, pochylenie, podkreślenie)
- Zmiana rozmiaru tabeli, ilości kolumn i wierszy

Zespół zdecydował się wykorzystać wzorzec architektoniczny *MVC*. Celem tego było wyszczególnienie trzech warstw aplikacji:

- Model - zajmująca się danymi aplikacji,
- Widok - zajmująca się warstwą widoku aplikacji - wyświetlaniem podstawowych danych i komunikacją z użytkownikiem
- Kontroler - pośredniczy między widokiem a modelem i informuje o zmianach stanu.

2 Praca nad projektem

2.1 Dobór narzędzi

Sam program stworzony został w języku Python - jest to język, z którym większość zespołu miała styczność już wcześniej i znała jego podstawy, co pozwoliło do sprawnego przejścia do części właściwej projektu.

W projekcie wykorzystane zostały dwie dodatkowe biblioteki:

- Tkinter - biblioteka ułatwiająca tworzenie Graficznego Interfejsu Użytkownika.
- Pytest - biblioteka pomagająca w tworzeniu automatycznych testów jednostkowych i tworzeniu raportów na temat procentowego pokrycia kodu testami.

Poza tymi ściśle programistycznymi narzędziami, zespół wybrał kilka rozwiązań dla ułatwienia pracy zespołowej, organizacji pracy, spisywania godzin. Są to kolejno:

- GitHub - na tej platformie stworzone zostało repozytorium, które następnie każdy członek zespołu sklonował, a następnie umieszczane były na nim kolejne poprawki, testy, usprawnienia i nowe funkcjonalności programu.
- Microsoft Teams - zespół starał się spotykać regularnie co tydzień by omówić bieżące sprawy, problemy oraz zaproponować i przeprowadzić dyskusję i/lub głosowanie na temat rozwiązań.
- Arkusz kalkulacyjny - wykorzystywany do zapisywania godzin poświęconych na poszczególne zadania w projekcie przez każdego z członków. Umieszczony został w zespole na platformie Teams

2.2 Wyszczególnienie zadań

Na początku wspólnej pracy zespół dokonał rozpoznania zadań, które należałoby wykonać, aby dostarczyć działający i spełniający założenia końcowy produkt. Zadania te to między innymi:

1. Spełnienie wymagań wstępnych
 - Rozpoznanie sposobu tworzenia tabel w LaTeX
 - Dobór narzędzi i bibliotek
 - Nauka korzystania z wybranych bibliotek i narzędzi
2. Tworzenie oprogramowania
 - Stworzenie warstwy widoku
 - Stworzenie modelu
 - Stworzenie kontrolera
 - Napisanie modułu eksportującego tabelę do kodu LaTeX
 - Napisanie testów jednostkowych
 - Testowanie manualne
3. Spełnienie wymogów projektowych
 - Tworzenie prezentacji
 - Skomponowanie dokumentacji projektu
 - Nagranie filmu przedstawiającego działanie programu

3 Efekt końcowy

3.1 Gdzie dostępny jest wynik końcowy

- Film demonstrujący efekt prac zespołu dostępny jest na platformie YouTube pod linkiem [https : //youtu.be/dtCqh_eWXM](https://youtu.be/dtCqh_eWXM)
- Repozytorium dostępne na platformie GitHub pod linkiem: <https://github.com/BialekPL/Tabelkomistrz>

3.2 Instrukcja obsługi programu

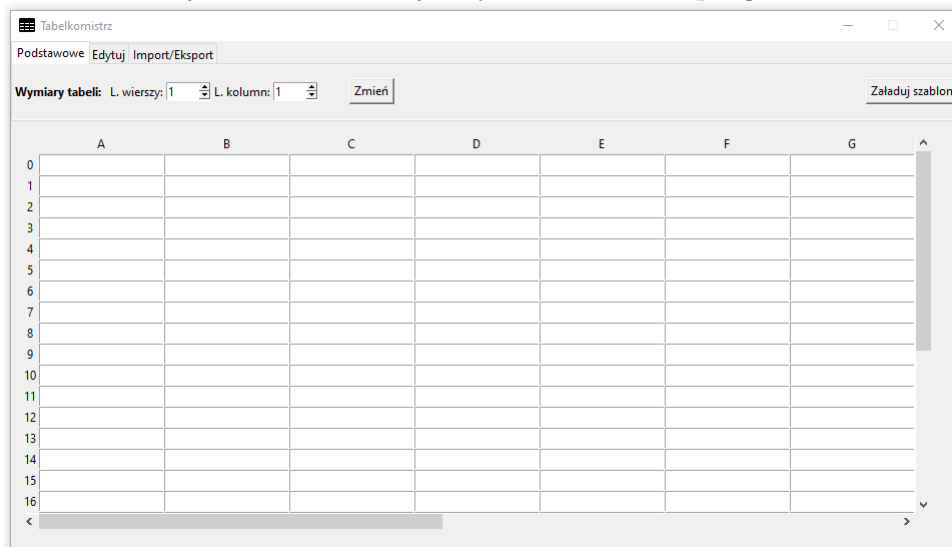
Wymagania do uruchomienia programu:

- Python 3.9+
- zainstalowana biblioteka Tkinter

Będąc w folderze głównym projektu uruchamiamy program za pomocą polecenia

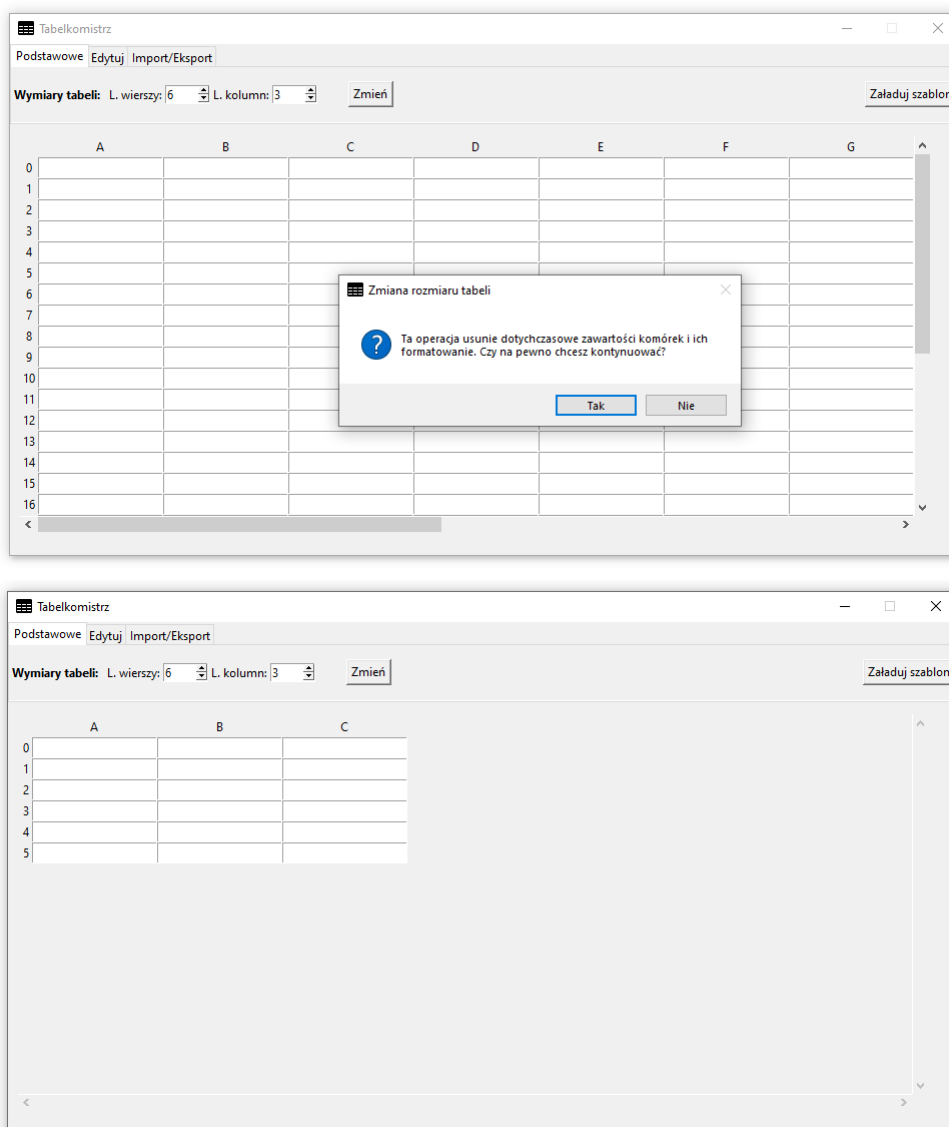
> python app.py

Oczom użytkownika ukazuje się okno startowe programu.





3.2.1 Zmiana rozmiaru tabeli

W celu zmiany rozmiaru tabeli w zakładce *Podstawowe* wpisujemy żadaną liczbę wierszy i kolumn, po czym naciskamy przycisk *Zmień*. Akceptujemy nasz wybór w wyświetlonym komunikacie i rozmiar tabeli ulega zmianie na wprowadzony.



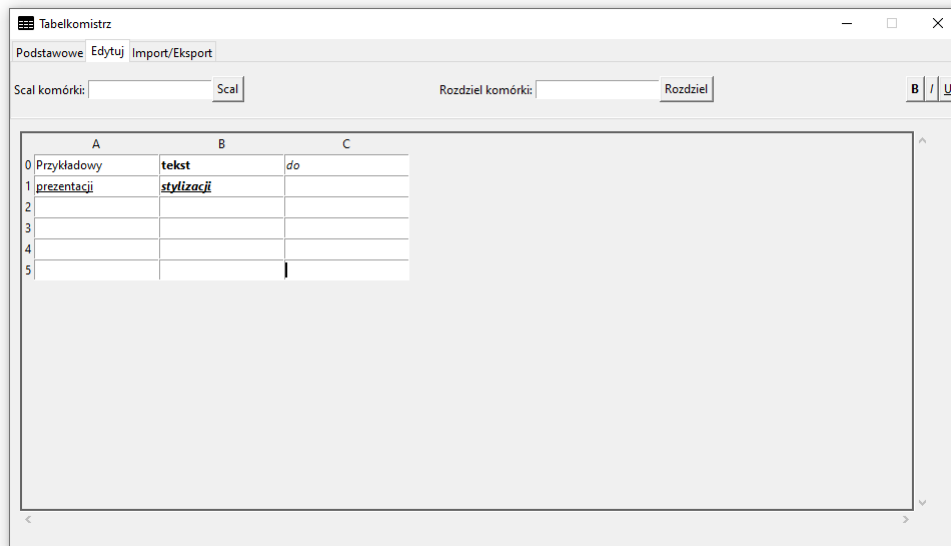
3.2.2 Stylizacja tekstu w komórkach

Do stylizacji tekstu w poszczególnych komórkach służy zakładka *Edytuj*. Wybierając jedną z ikon w prawym górnym rogu zmieniamy styl obecnie wybranej komórki. Dostępne opcje stylizacji:

-  - pogrubienie tekstu
-  - pochylenie tekstu

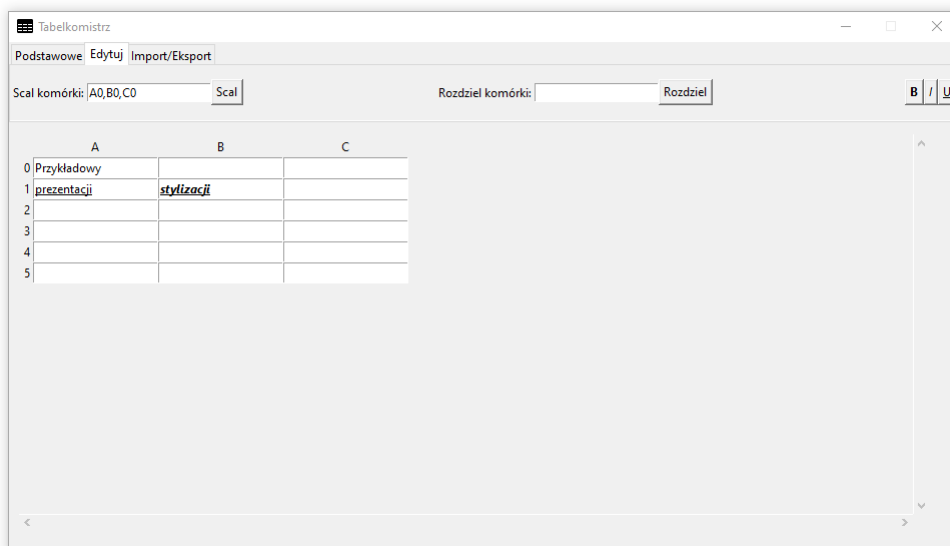
- U - podkreślenie tekstu

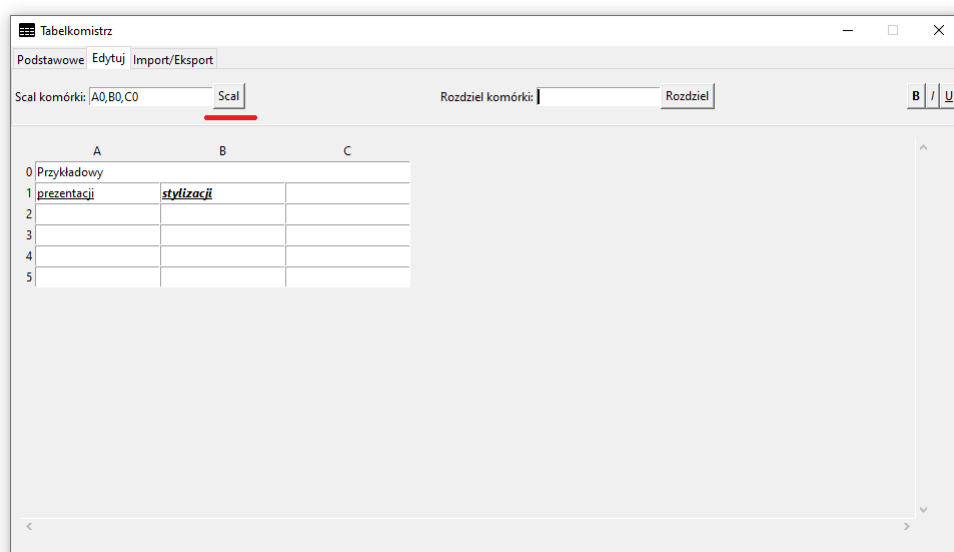
Stylizacje można łączyć, co zostało zaprezentowane w poniższym przykładzie:



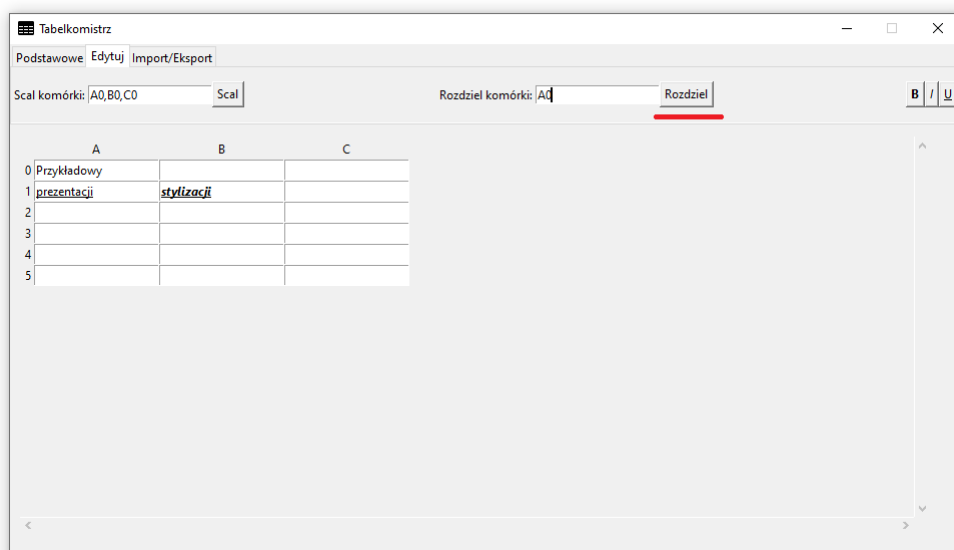
3.2.3 Scalanie i rozdzielanie komórek

W tej samej zakładce dokonać można scalenia i rozdzielania komórek. W polu tekstowym umieścić należy indeksy komórek oddzielone przecinkami, a następnie kliknąć przycisk Scal.



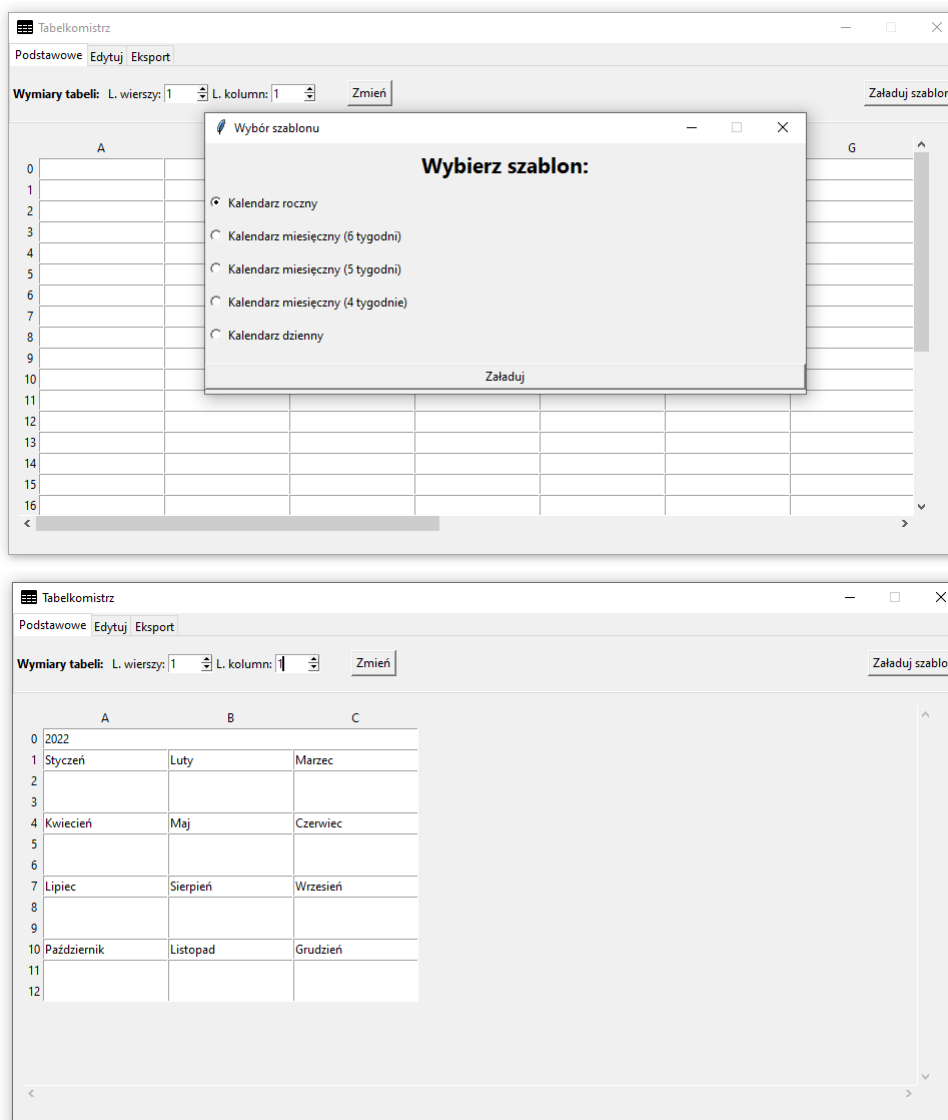


By rozdzielić komórki należy w odpowiednim polu tekstowym wpisać indeks jednej z komórek, które należą do scalonej grupy i kliknąć przycisk Rozdziel.



3.2.4 Szablony

W programie przygotowane zostało kilka przykładowych szablonów do wykorzystania. Dostęp do nich znajduje się w prawym górnym rogu w postaci przycisku *Wybierz szablony*. Po jego naciśnięciu pojawia się lista z szablonami do wyboru.



3.2.5 Eksport

Aby wyeksportować tabelę do kodu LaTeX należy w zakładce Eksport nacisnąć przycisk eksportuj. Stworzony zostanie w katalogu głównym plik *eksport.txt* z gotowym kodem.

4 Podsumowanie

Podstawowe założenia projektu zostały spełnione. Zaimplementowane zostały nawet dodatkowe funkcjonalności, między innymi stylizowanie zawartości komórek.

Grupa rozwinęła się nie tylko pod względem ściśle programistycznym, ale również pod względem pracy w grupie, wspólnie naradzając się nad spisywanymi do wykonania zadaniami, spotykając się cotygodniowo na omówienie bieżących spraw i problemów, wynajdując rozwiązania na kolejno pojawiające się przeciwności.

4.1 Dalszy rozwój

Projekt oczywiście można dalej rozwijać, dokładając do niego kolejne funkcjonalności, (jak na przykład importowanie tabeli do programu z kodu LaTeX), lub poprawiając wydajność i wygodę użytkowania. Na dzień dzisiejszy jednak, zespół zadowolony jest z aktualnego stanu projektu.

5 Bibliografia

- [1] Python community. „Dokumentacja biblioteki Pytest”. W: *PYTEST DOCS* (2021). URL: <https://docs.pytest.org/en/6.2.x>.
- [2] Python community. „Dokumentacja biblioteki Tkinter”. W: *PYTHON DOCS* (2021). URL: <https://docs.python.org/3/library/tkinter.html>.