|  |  |
| --- | --- |
| PS **1** | Rola |
| Pfajfer Jakub | Kierownik |
| Kiełczewski Krzysztof | …. |
| Samluk Mateusz | …. |
| Polejczuk Adam | …. |

**Tygodniowy plan działań – TYDZIEŃ 2 (10/08/2020)**

Temat projektu: Implementacja prostego serwisu webowego w technologii Django do zarządzania zadaniami wyznaczonymi przez nextflow.

|  |  |  |  |
| --- | --- | --- | --- |
| lp | Zadanie do zrobienia | Uwagi | Os. odpowiedzialna |
| 1 | Opis projektu | - | Polejczuk Adam |
| 2 | Diagram klas | - | Polejczuk Adam |
| 3 | Podstawowe wymagania klienta | - | Pfajfer Jakub |
| 4 | Przykładowe scenariusze przypadków użycia | - | Pfajfer Jakub |
| 5 | Przygotowanie repo | - | Samluk Mateusz |
| 6 | Mockup interfejsu | - | Samluk Mateusz |
| 7 | Diagram przypadków użycia | - | Kiełczewski Krzysztof |
| 8 | Zaprojektować modele w Django | - | Kiełczewski Krzysztof |

Wykonane zadania:

|  |  |  |  |
| --- | --- | --- | --- |
| lp | Zadanie do zrobienia | Wykonany | Os. odpowiedzialna |
| 1 | Opis projektu |  | Polejczuk Adam |
| 2 | Diagram klas |  | Polejczuk Adam |
| 3 | Podstawowe wymagania klienta |  | Pfajfer Jakub |
| 4 | Przykładowe scenariusze przypadków użycia |  | Pfajfer Jakub |
| 5 | Przygotowanie repo |  | Samluk Mateusz |
| 6 | Mockup interfejsu |  | Samluk Mateusz |
| 7 | Diagram przypadków użycia |  | Kiełczewski Krzysztof |
| 8 | Zaprojektować modele w Django |  | Kiełczewski Krzysztof |

Opis projektu:

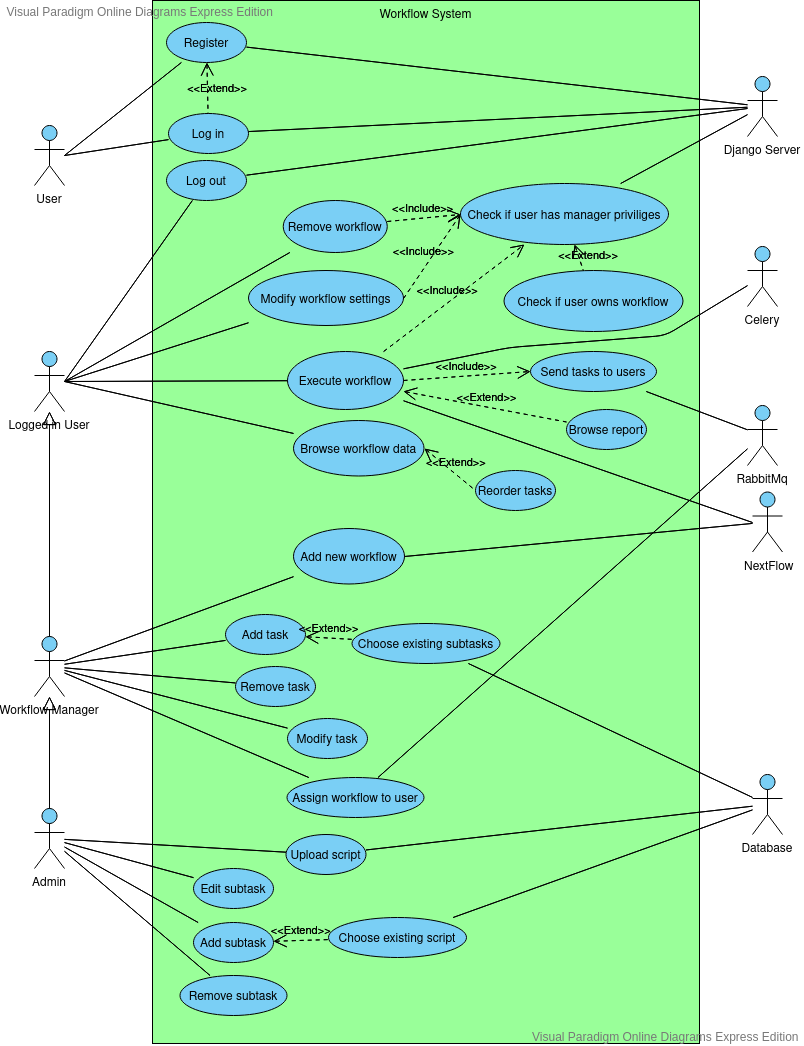
Zadaniem jest tej aplikacji jest zarządzanie (dodawanie, modyfikowanie, usuwanie, uruchamianie) workflow. Każdy workflow składa się z pomniejszych tasków, w których w skład wchodzą subtaski. Subtaski są poszczególnymi skryptami, które zapisane są w formacie .sh lub .py. Nextflow zostanie wykorzystany do tworzenia i uruchamiania workflow. Celery oraz RabbitMQ zostanie wykorzystany do równoległego uruchamiania skryptów Nextflow. Aplikacja będzie napisana w języku Python z wykorzystaniem framework’a Django. W aplikacji występowały będą trzy role użytkowników: użytkownik, zarządca workflow oraz administrator. Każda z ról posiada własne uprawnienia:

* Użytkownik może jedynie uruchomić workflow oraz przejrzeć jego wyniki,
* Zarządca workflow ma uprawnienia użytkownika, dodatkowo może zarządzać workflow i taskami,
* Administrator posiada uprawnienia zarządcy workflow. Poza tym, może zarządzać subtaskami oraz dodawać skrypty.

Podstawowe funkcjonalności:

* Zarządzanie subtaskami, taskami (dodawanie, modyfikowanie, usuwanie),
* Zarządzanie workflow (dodawanie, modyfikowanie, usuwanie, uruchamianie, ustalanie kolejności tasków oraz subtasków),
* Dodawanie skryptów w postaci plików (.py i .sh),
* Przeglądanie wyników workflow i każdej z jego części,
* Logowanie i rejestracja,
* Uruchomienie workflow dla wielu użytkowników jednocześnie,

Diagram przypadków użycia:



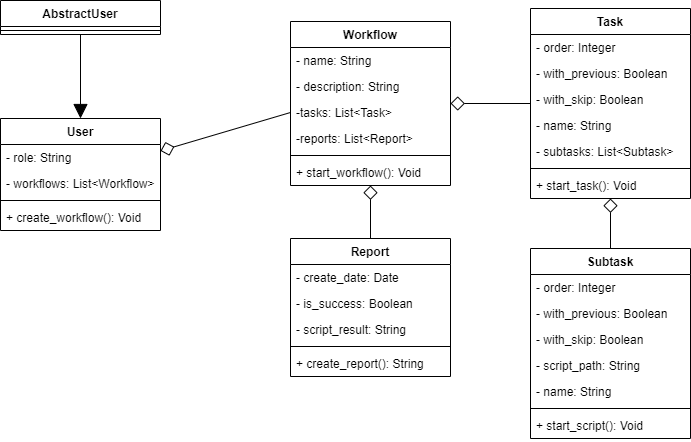
Scenariusze przypadków użycia:

|  |  |
| --- | --- |
| Scenariusz 1: Użytkownik uruchamia i przegląda raport workflow | |
| Aktorzy: | Użytkownik |
| Przebieg zdarzeń: | * Użytkownik loguje się do systemu, * Użytkownik przegląda listę dostępnych workflow i wybiera ten, który chce uruchomić, * Użytkownik czeka na wykonanie wszystkich tasków danego workflow, * Po zakończeniu wszystkich tasków, użytkownik ma możliwość podejrzenia raportu. |
| Wyniki przebiegu: | * Raport przebiegu workflow |
| Częstotliwość wykonywania: | * Kilka razy dziennie |
| Sytuacje wyjątkowe: | * Użytkownik może zatrzymać bądź przerwać wykonywanie workflow, * Użytkownik może podać niewłaściwe dane do logowania, |

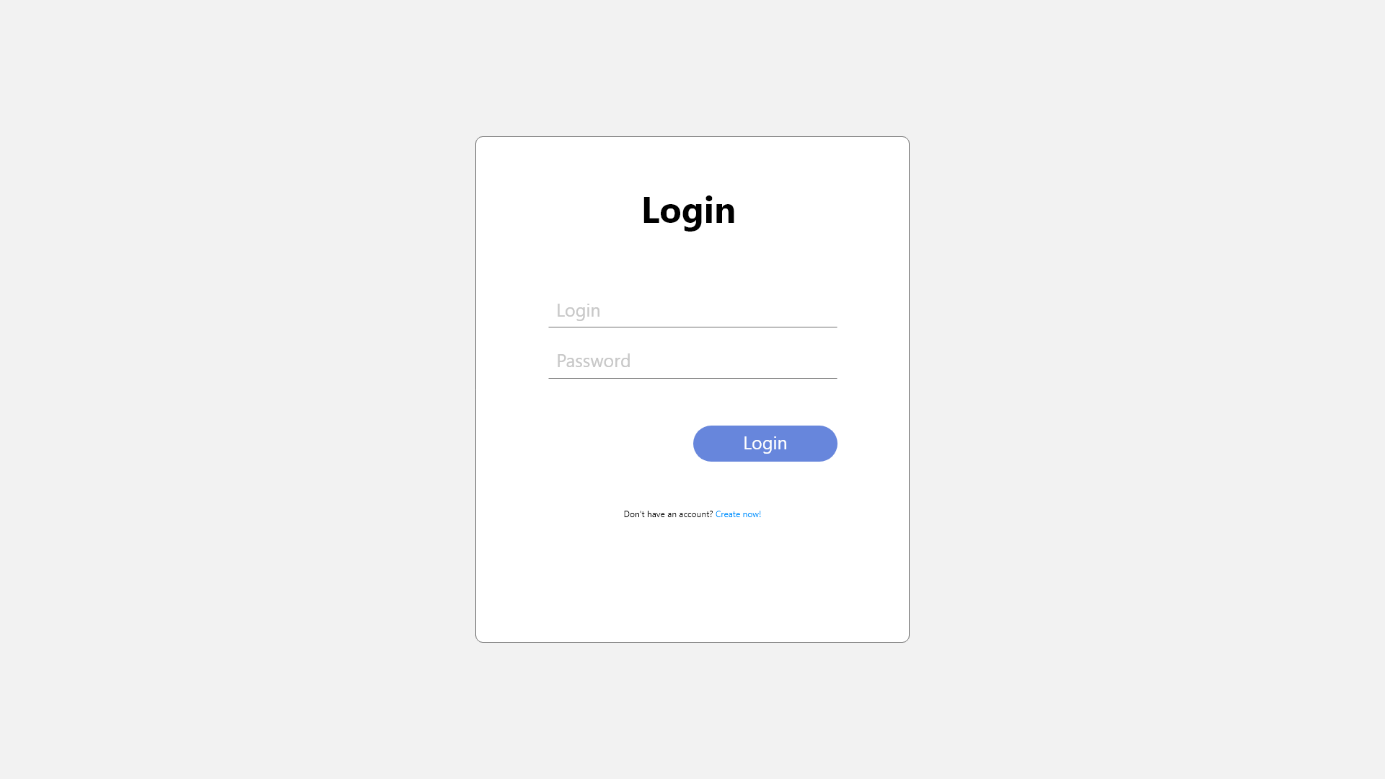
|  |  |
| --- | --- |
| Scenariusz 2: Zarządzanie workflow | |
| Aktorzy: | Zarządca workflow |
| Przebieg zdarzeń: | * Użytkownik loguje się do systemu, * Użytkownik może uruchomić istniejące już workflow, * Użytkownik może modyfikować bądź usunąć istniejące taski, * Użytkownik może utworzyć taska z udostępnionych mu przez administratora subtasków, * Użytkownik może stworzyć workflow z dostępnych tasków, * Użytkownik może ustalić kolejność wykonywania tasków i czy powinny być uruchamiane równocześnie, * Użytkownik może przypisać dany workflow do danych użytkowników, |
| Wyniki przebiegu: | * Dodanie nowych workflow do listy * Dodanie nowych tasków do listy |
| Częstotliwość wykonywania: | * Kilka razy tygodniowo |
| Sytuacje wyjątkowe: | * Użytkownik może zatrzymać bądź przerwać wykonywanie workflow, * Użytkownik może podać niewłaściwe dane do logowania, * Użytkownik nie posiada uprawnień zarządcy workflow, |

|  |  |
| --- | --- |
| Scenariusz 3: Zarządzanie subtaskami | |
| Aktorzy: | Administrator |
| Przebieg zdarzeń: | * Użytkownik loguje się do systemu, * Użytkownik może wgrać skrypt, który posiada format .py bądź .sh, * Użytkownik może edytować bądź usunąć istniejące już subtaski, * Dodatkowo, ma możliwość zrobienia wszystkiego co mogą zrobić pozostałe role. |
| Wyniki przebiegu: | * Dodany zostanie subtask do listy subtasków |
| Częstotliwość wykonywania: | * 1-2 razy tygodniowo po wstępnym dodaniu dużej ilości do bazy, |
| Sytuacje wyjątkowe: | * Użytkownik może nie posiadać wystarczających uprawnień, * Użytkownik może wgrać niepoprawnie napisany skrypt, * Użytkownik może podać niewłaściwe dane do logowania, |

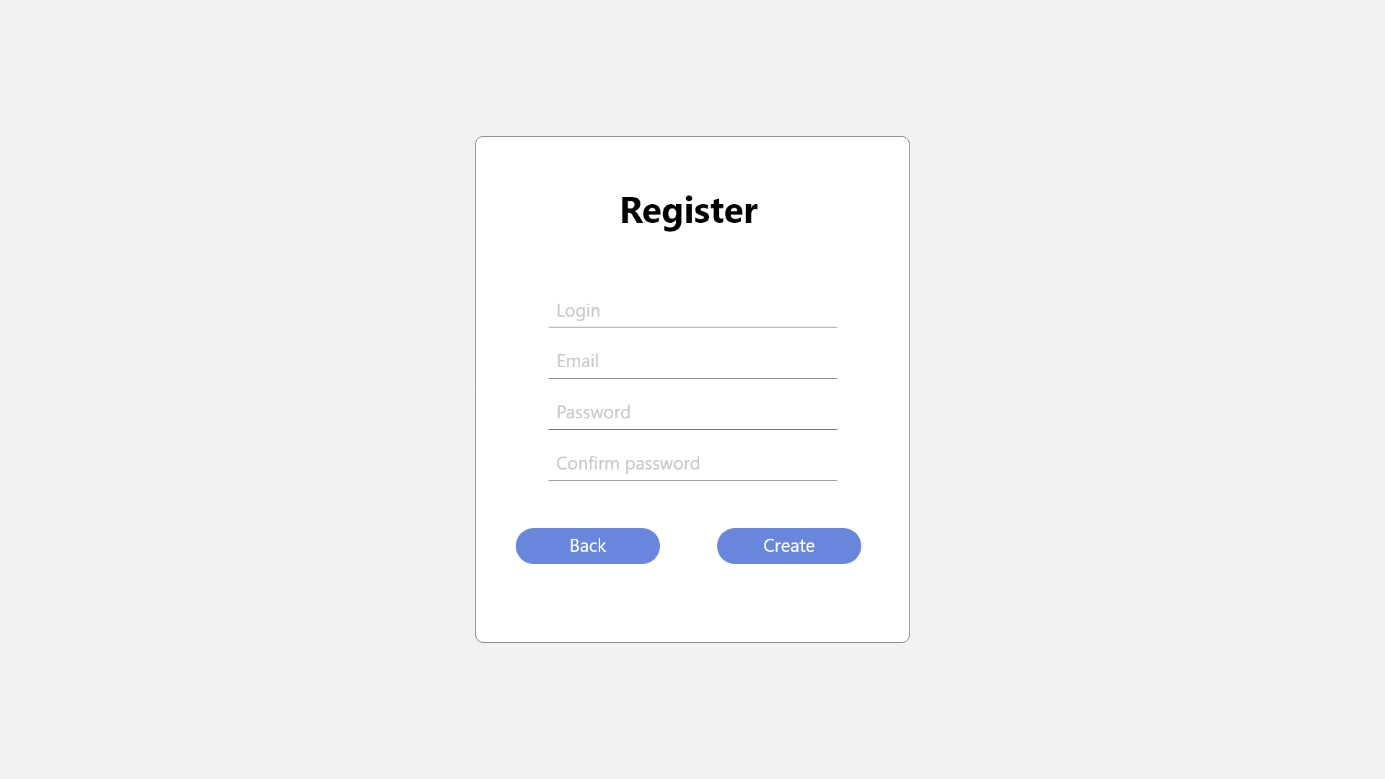
Diagram klas:



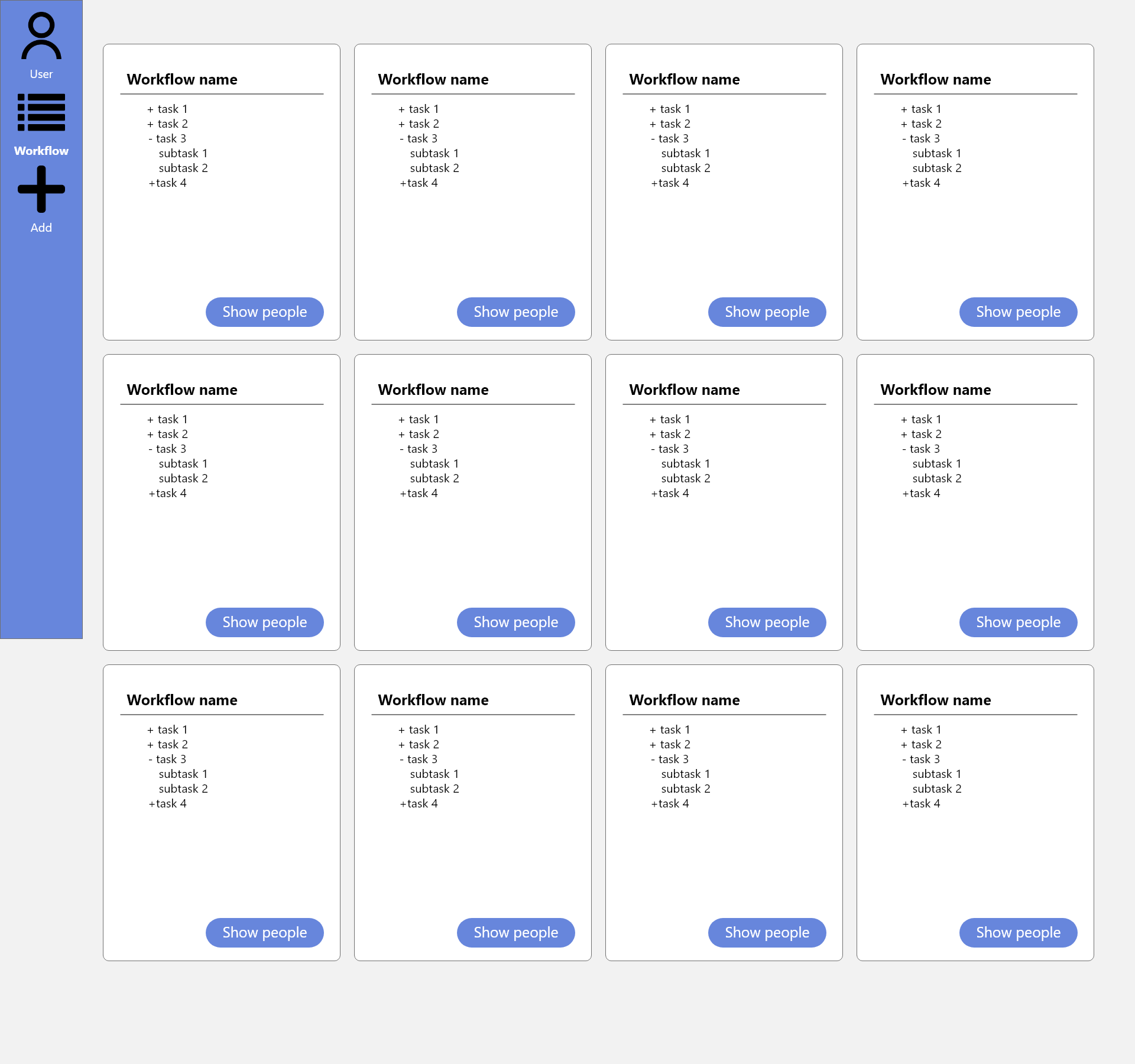
Mockup:



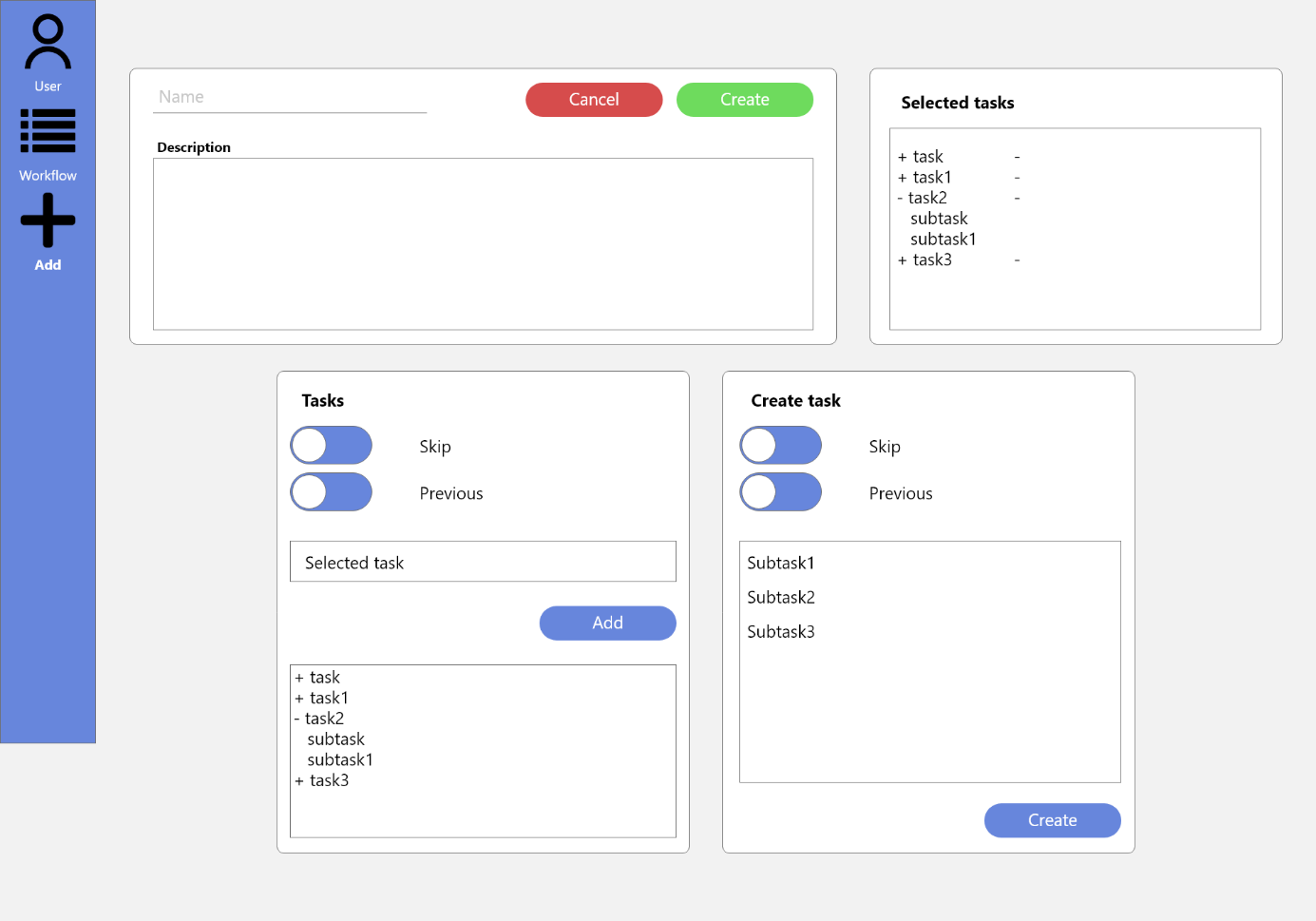
Rysunek 1 Okno logowania



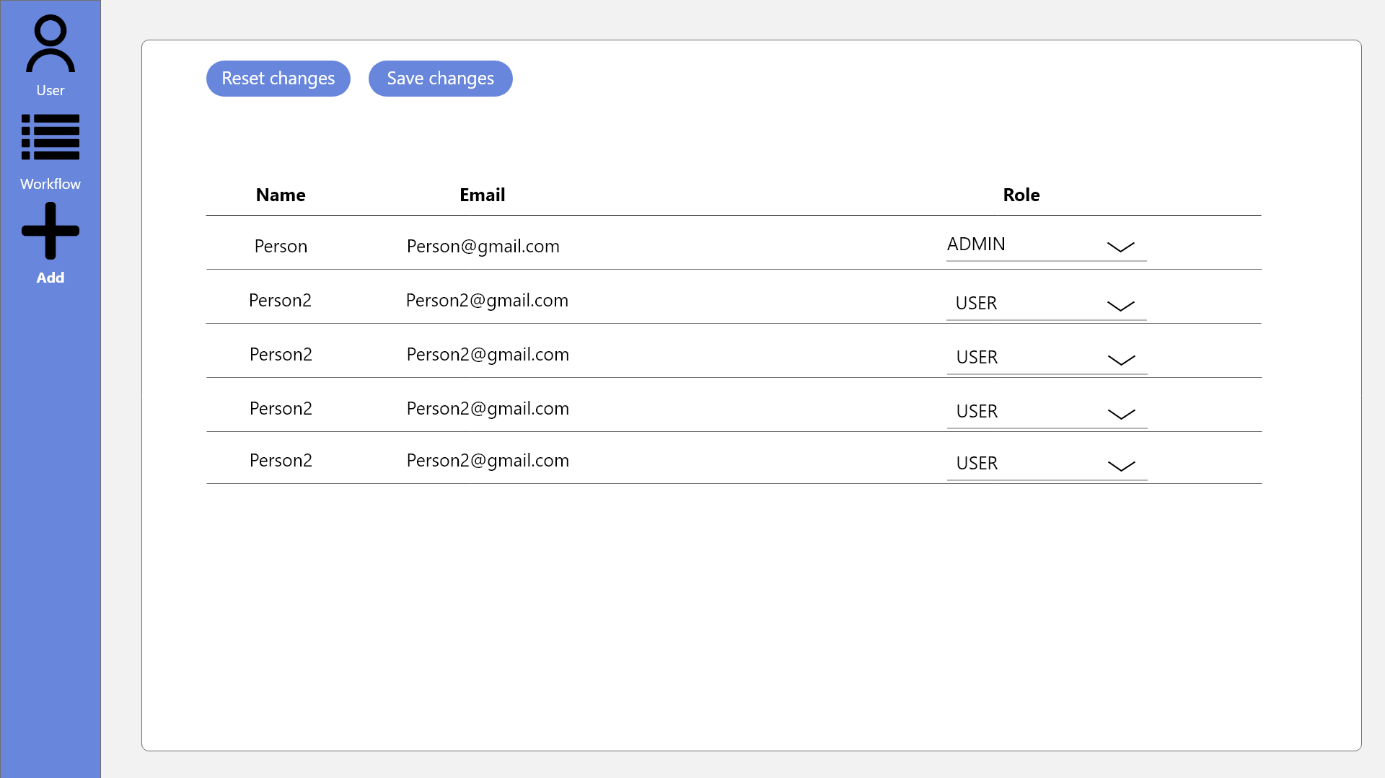
Rysunek 2 Okno rejstracji



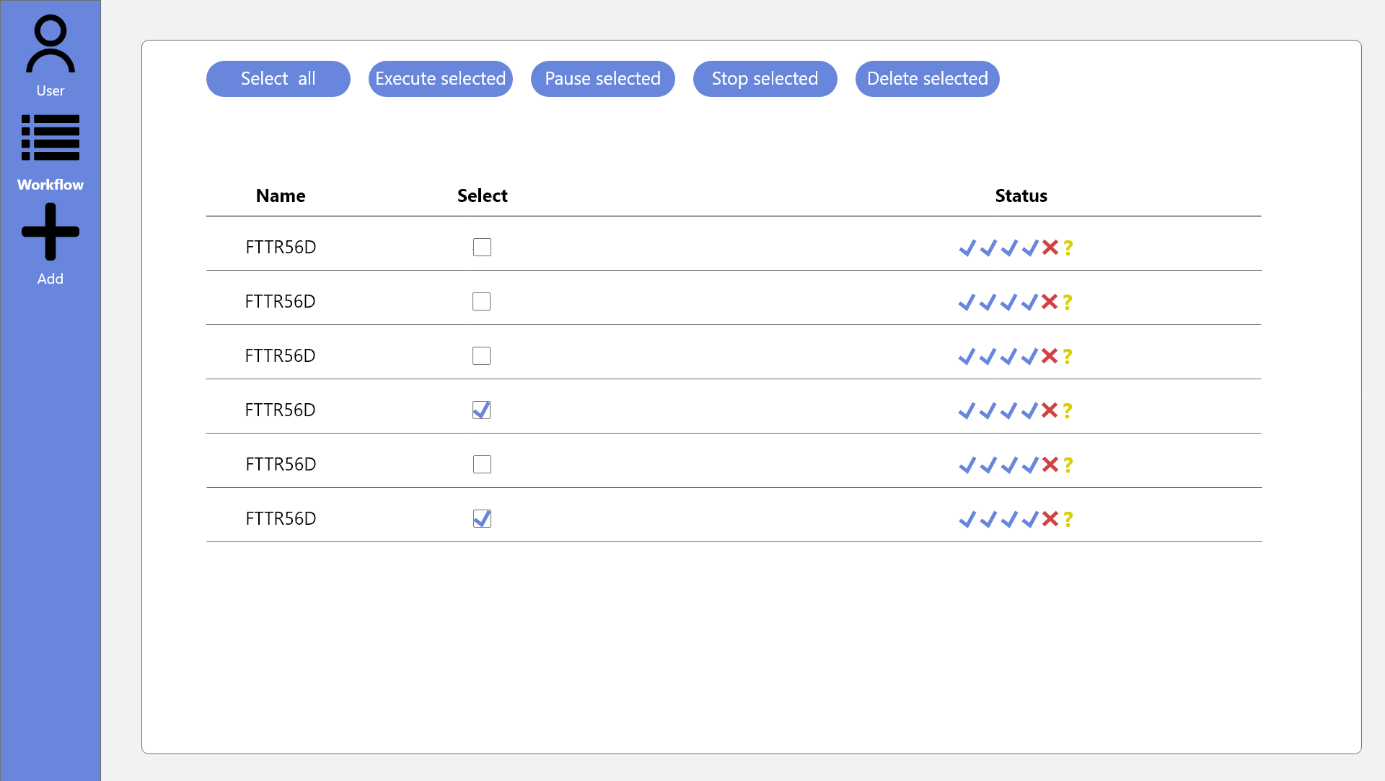
Rysunek 3 Okno widoku wszystkich workflowów



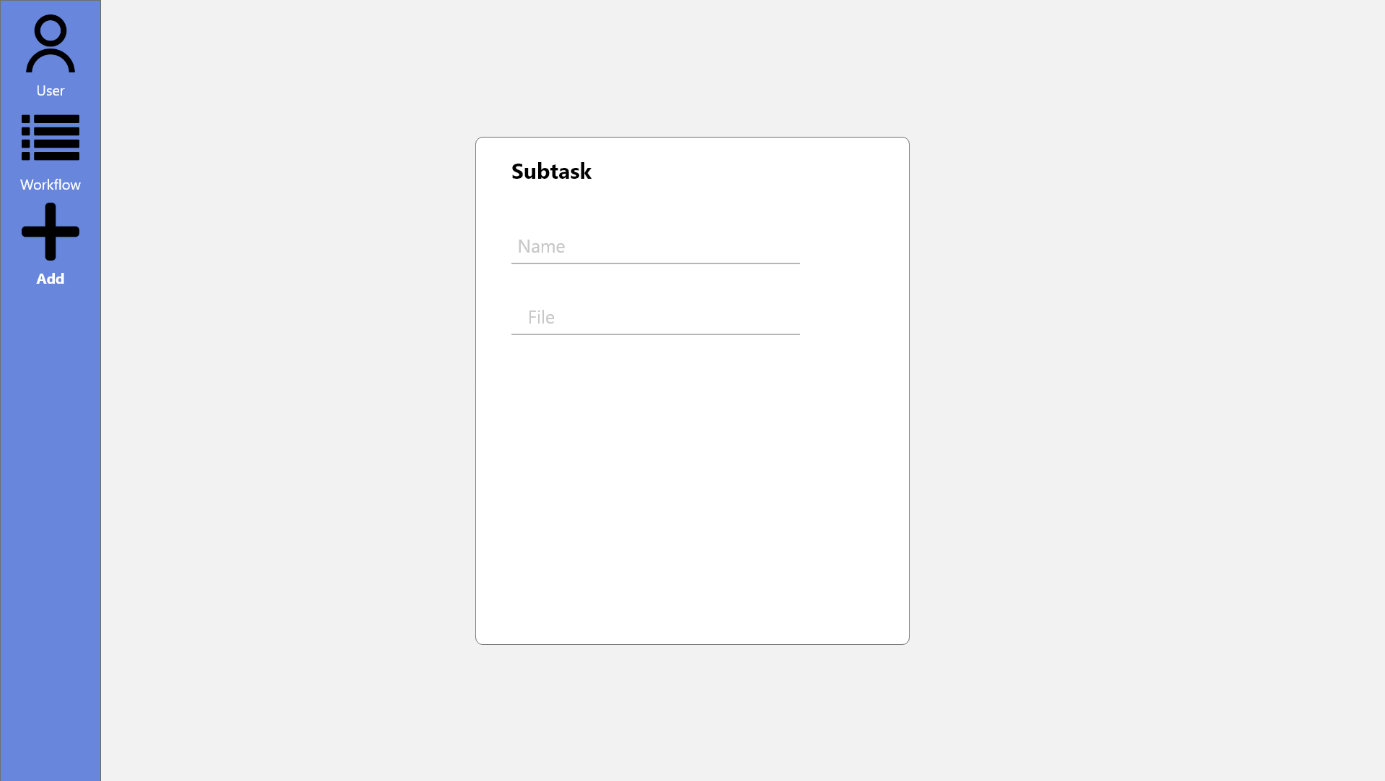
Rysunek 4 Okno dodawania nowych workflowów i nowych tasków



Rysunek 5 Okno zarządzania użytkownikami



Rysunek 6 Okno uruchamiania workflowów



Rysunek 7 Okno dodawania nowych subtasków