

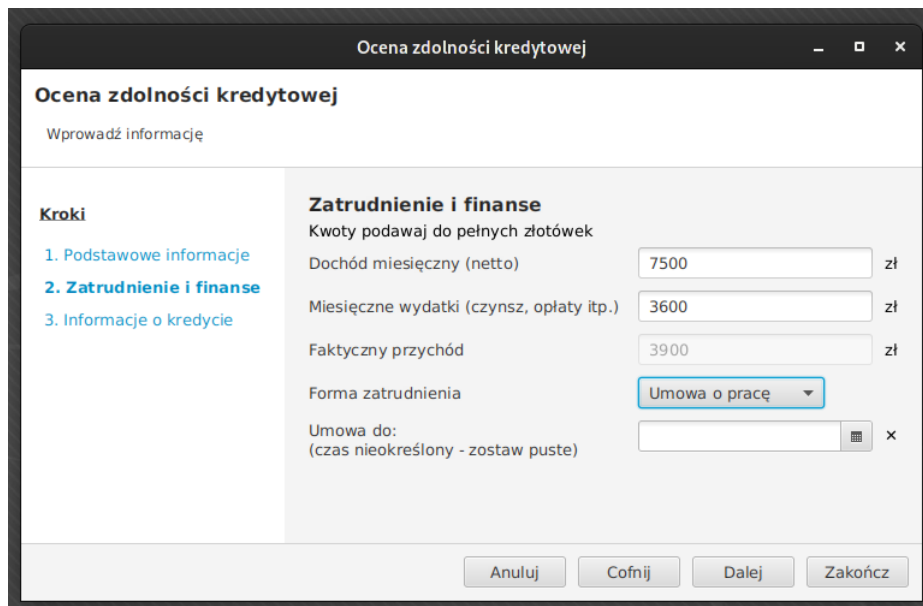
# System oceny zdolności kredytowej

## Laboratorium Systemów Rozmytych

Patryk Dolata 460930

### 1 Raport postępu prac

Tak jak wspominałem w planie projektu, do tworzenia mojego systemu wykorzystuję język `Kotlin` i framework `TornadoFX`. Na ten moment jest to aplikacja desktopowa w stylu "Wizard". Tzn. są oddzielne strony, na których użytkownik po kolei wprowadza swoje informacje.



Rysunek 1: Jedna ze stron w formularzu

Na każdej stronie ("Kroki" po lewej stronie na Rys. 1) znajduje się formularz, który użytkownik musi wypełnić. Na podstawie danych podanych przez użytkownika w formularzu będzie obliczany *credit score* tego użytkownika. Postanowiłem również cały mój problem rozdzielić na podproblemy. Wszystkie zmienne składowe podzieliłem sobie na 2-3 grupy:

- **analiza jakościowa**, opisująca kredytobiorcę: wiek, wykształcenie, stan cywilny, staż pracy
- **analiza ilościowa/finansowa**, opisująca kondycję finansową kredytobiorcy: dochody, miesięczne wydatki, forma zatrudnienia, data wygaśnięcia umowy
- **analiza kredytu**, który kredytobiorca chce wziąć: kwota, wkład własny, oprocentowanie, liczba rat

Prawdopodobnie w późniejszym czasie zrezygnuje z grupy **analizy kredytu**, gdyż same informacje o tym jaki kredyt chcemy wziąć nie dadzą żadnej sensownej informacji do końcowego wyniku nie znając całego kontekstu (dochodów kredytobiorcy itd.). Najprawdopodobniej grupa ta zostanie połączona z **analizą finansową**, więc oprócz sprawdzania kondycji finansowej, analiza ta będzie również sprawdzała czy kredytobiorca jest w ogóle w stanie wziąć taki kredyt (jeśli np. rata kredytu będzie większa niż *zarobki – wydatki* użytkownika, no to score prawdopodobnie będzie bliski zeru).

Każda z analiz docelowo ma dawać w wyniku zmienną lingwistyczną w przedziale [0-100] - mały, średni, wysoki. Następnie wyniki analiz wykorzystujemy do uzyskania końcowego wyniku tzn. *credit score*, który również będzie zmienną lingwistyczną w przedziale [0-100] - mały, średni, wysoki.

Takie rozdzielenie całego problemu na mniejsze podproblemy pozwoli na uniknięcie ogromnej bazy reguł składającej się ze wszystkich zmiennych naraz i rozłożenie tego na pomniejsze, uporządkowane bazy reguł.

Z technicznych postępów to mogę powiedzieć, że wszystkie 3 formularze są już gotowe (należałoby je tylko w późniejszym stanie ostrylować). Wstępne pliki `.fcl` dla analiz już stworzyłem, głównym zadaniem, które zostało jest stworzenie baz reguł dla poszczególnych wnioskowań rozmytych. Logikę rozmytą za pomocą biblioteki `jFuzzyLogic.jar` już mam zaimplementowaną, więc po stworzeniu baz reguł wnioskowania pozostanie tylko wyświetlić użytkownikowi stosowny wynik.