FirstHome Advisor

Aplikacja wspierająca młodych dorosłych (w wieku 25–35 lat) w ocenie ich gotowości finansowej do zakupu pierwszego mieszkania z wykorzystaniem kredytu hipotecznego. Aplikacja pełni rolę doradczą i edukacyjną, bazując na danych użytkownika oraz modelu klasyfikacyjnym, który analizuje zdolność kredytową i ryzyko.

# Grupa docelowa

Osoby młode planujące zakup pierwszego mieszkania

Użytkownicy z dochodami z umów o pracę, działalności gospodarczej, freelancingu

Osoby bez wiedzy finansowej, które chcą lepiej zrozumieć swoje możliwości

Użytkownicy zainteresowani samodzielną oceną zdolności kredytowej

# Główne funkcjonalności

## MVP

* Formularz do wprowadzania danych osobistych i finansowych (wiek, dochód, forma zatrudnienia, miesięczne zobowiązania, wkład własny, itp.)
* Klasyfikacja użytkownika jako gotowego, prawie gotowego lub niegotowego do kredytu
* Wyliczenie wskaźników (DTI itp.)
* Krótkie, spersonalizowane zalecenia na podstawie klasyfikacji
* Przejrzysta prezentacja wyników i ryzyka

## Wersja rozszerzona (opcjonalna):

* Symulacja “co jeśli?” — analiza zmian w dochodach, wkładzie własnym, zobowiązaniach
* Historia i zapisywanie scenariuszy użytkownika
* Wybór różnych typów kredytów i analizowanie wpływu parametrów (oprocentowanie, okres spłaty)
* Możliwość porównania z danymi statystycznymi (średni DTI, średnie raty w danym regionie)
* Generowanie raportu PDF z analizą

# Technologie

* Język programowania: Python
* Biblioteki: Pandas, Scikit-learn, Matplotlib, Streamlit
* Model ML: klasyfikator do oceny ryzyka
* Możliwość wykorzystania SHAP lub innego narzędzia do interpretacji predykcji

# Zakres danych wejściowych

* Wiek użytkownika
* Rodzaj zatrudnienia i stabilność dochodu
* Wysokość dochodu miesięcznego
* Miesięczne zobowiązania finansowe (np. raty, karty kredytowe, leasing)
* Wysokość wkładu własnego
* Cena planowanego mieszkania
* Liczba osób na utrzymaniu

# Zakres danych wyjściowych

* Ocena gotowości kredytowej (np. gotowy / wymaga poprawy / niegotowy)
* Wskaźnik ryzyka (np. prawdopodobieństwo klasyfikacji jako klient ryzykowny)
* Wartości wybranych wskaźników: DTI, udział wkładu własnego, prognozowana rata
* Indywidualne sugestie poprawy zdolności kredytowej

# Interfejs użytkownika

* Webowa aplikacja dostępna z poziomu przeglądarki
* Intuicyjny formularz do wprowadzania danych
* Sekcja z wizualizacją wyników i wskazówkami
* Możliwość zapisania i porównania różnych scenariuszy

# Możliwości rozwoju

Integracja z bazą danych bankowych lub API porównywarek kredytowych

Rozszerzenie o scoring kredytowy na podstawie rzeczywistych danych BIK (symulacyjnie)

Wersja mobilna lub PWA

Uwzględnienie wskaźników ekonomicznych i inflacyjnych w analizach

Opcja tworzenia konta użytkownika i długofalowego śledzenia sytuacji

# Klasy

##### UserData

Opis: reprezentuje dane finansowe i osobiste wprowadzone przez użytkownika.

Atrybuty:

* age: int
* employment\_type: str
* monthly\_income: float
* monthly\_expenses: float
* existing\_loans: float
* own\_contribution: float
* property\_value: float
* dependents: int

Metody:

to\_dataframe() -> pd.DataFrame – przygotowanie danych do klasyfikatora

##### RiskClassifier

Opis: opakowanie modelu ML do oceny ryzyka użytkownika.

Atrybuty:

* model – np. obiekt scikit-learn
* preprocessor – pipeline / scaler / encoder

Metody:

predict(user\_data: UserData) -> str – zwraca klasyfikację (np. "gotowy")

predict\_proba(user\_data: UserData) -> float – zwraca prawdopodobieństwo ryzyka

##### CreditCalculator

Opis: wykonuje obliczenia pomocnicze na podstawie danych użytkownika.

Metody:

calculate\_dti(user\_data: UserData) -> float

calculate\_max\_credit(user\_data: UserData) -> float

recommendations(user\_data: UserData) -> list[str]

##### AssessmentResult

Opis: struktura wyniku oceny.

Atrybuty:

* risk\_category: str
* risk\_score: float
* max\_loan: float
* dti: float
* recommendations: list[str]

##### AppController

Opis: koordynator logiki aplikacji, spina komponenty i zwraca gotowy wynik do UI.

Metody:

run\_assessment(user\_data: UserData) -> AssessmentResult

##### MainUI

Opis: warstwa interfejsu

Metody:

render\_form() -> UserData

display\_result(result: AssessmentResult)

# Wzorzec architektoniczny: MVC z logiką ML w serwisie

Model – UserData, AssessmentResult, dane ML (RiskClassifier)

Service – CreditCalculator, AppController

View/Interface – MainUI

# Przepływ danych

Użytkownik wypełnia formularz → MainUI tworzy UserData

AppController.run\_assessment():

używa RiskClassifier do predykcji

używa CreditCalculator do obliczeń

tworzy AssessmentResult

MainUI prezentuje wynik użytkownikowi

# Diagram UML

Obraz zawierający tekst, diagram, Czcionka, paragon

Opis wygenerowany automatycznie