

Uniwersytet Mikołaja Kopernika  
Wydział Matematyki i Informatyki

Paweł Marcin Chojnacki  
nr albumu: 260082  
informatyka

Praca inżynierska

# **Kopra – aplikacja mobilna wspomagająca inwestowanie w serwisie Kokos**

Opiekun pracy dyplomowej  
dr Jerzy Białkowski

Toruń 2016



# Spis treści

<b>Słownik pojęć</b>	<b>5</b>
<b>Wstęp</b>	<b>7</b>
Pomysł napisania aplikacji o tematyce finansowej . . . . .	7
Podobne rozwiązania w świecie aplikacji mobilnych . . . . .	8
Rozważane formy wdrożenia projektu . . . . .	8
Jaki jest cel aplikacji? . . . . .	8
Założenia projektowe . . . . .	9
Korzystanie z serwisów pożyczkowych na smartfonach . . . . .	9
<b>1. O pożyczaniu</b>	<b>11</b>
1.1. Krótka historia pieniądza . . . . .	11
1.2. Idea pożyczek społecznościowych . . . . .	11
1.3. Jakie są ryzyka związane z tym instrumentem finansowym? . . . . .	12
1.4. Jakie zyski można osiągnąć? . . . . .	12
1.5. Porównanie najpopularniejszych instrumentów finansowych . . . . .	13
1.6. Porównanie polskich serwisów pożyczkowych . . . . .	13
1.7. Wybór systemu do implementacji . . . . .	14
1.8. WebAPI . . . . .	14
<b>2. Wykorzystane technologie i narzędzia</b>	<b>17</b>
2.1. Microsoft Visual Studio Community 2015 . . . . .	17
2.2. Język C# 5.0 . . . . .	17
2.3. .NET Framework . . . . .	18
2.4. Windows Phone 8.1 SDK . . . . .	18
2.5. Newtonsoft.Json . . . . .	19
2.6. Git i GitHub . . . . .	19
2.7. LinqPad . . . . .	20
2.8. ILSpy . . . . .	20
<b>3. Budowa projektu</b>	<b>21</b>
3.1. Wzorzec MVVM . . . . .	21
3.1.1. Opis wzorca Model View ViewModel . . . . .	21
3.1.2. MVVM vs MVC . . . . .	22
3.1.3. Implementacja MVVM w aplikacji . . . . .	22
3.2. Kokos WebAPI . . . . .	24
3.2.1. Opis interfejsu webowego . . . . .	24
3.2.2. Implementacja API w aplikacji . . . . .	25

3.3. Notyfikacje przez BackgroundTask . . . . .	26
3.3.1. Opis . . . . .	26
3.3.2. Implementacja NewAuctionNotifier . . . . .	26
<b>4. Prezentacja interfejsu aplikacji . . . . .</b>	<b>27</b>
4.1. Etymologia nazwy . . . . .	27
4.2. Korzyści względem wersji przeglądarkowej . . . . .	27
4.3. Koszt użytkowania aplikacji . . . . .	27
4.4. Bezpieczeństwo danych przechowywanych w aplikacji . . . . .	27
4.5. Jak wygląda Kopra? . . . . .	28
4.5.1. Logo . . . . .	28
4.5.2. Splash screen . . . . .	28
4.5.3. Strona logowania . . . . .	28
4.5.4. Strona główna . . . . .	28
4.6. Instrukcja obsługi do aplikacji . . . . .	29
<b>5. Podsumowanie . . . . .</b>	<b>31</b>
<b>Spis rysunków . . . . .</b>	<b>33</b>
<b>Bibliografia . . . . .</b>	<b>35</b>

# Słownik pojęć

## **Aktywa finansowe**

Środki finansowe przechowywane przez podmiot gospodarczy (osobę lub firmę), mające na celu przyniesienie w przyszłości korzyści ekonomicznych. Są to przechowywane środki pieniężne, umowy kredytowe, akcje innych firm.

## **Aukcja pożyczkowa**

Oferta umowy pożyczkowej wystawiona przez pożyczkobiorcę, zweryfikowana na określonej platformie zgodnie z jej kryteriami. Oferta taka skierowana jest do wielu podmiotów określonych jako *pożyczkodawców*.

## **Automat inwestycyjny**

Oprogramowanie przeznaczone do symulowania działań pożyczkodawcy przez wpłacanie środków na aukcje spełniające określone kryteria.

## **FriendlyScore**

Przyznanie punktacji użytkownikowi w oparciu o analizę danych z kont w serwisach społecznościowych, np. Facebooka czy Twittera oraz dodatkowe weryfikacje, np. weryfikację pracodawcy.

## **Giełda Papierów Wartościowych**

Jest to instytucja publiczna zapewniająca możliwość obrotu giełdowymi papierami wartościowymi. Pozwala ona na kupno oraz sprzedaż papierów wartościowych takich jak akcje i obligacje. W mojej pracy ograniczę się do polskiego rynku kapitałowego, dlatego pojęcie „Giełda Papierów Wartościowych” (w skrócie „Giełda”) będzie w niej oznaczać instytucję „Giełda Papierów Wartościowych w Warszawie SA”.

## **Instrument finansowy (*ang. financial instrument*)**

Jest to kontrakt zawarty między dwiema stronami regulujący zależność finansową między tymi stronami. W tym kontrakcie zawarte są dwie strony. Strona długa (*ang. long party*), inaczej – posiadacz długiej pozycji, oraz strona krótka (*short party*), inaczej – posiadacz krótkiej pozycji. Przykładem instrumentu finansowego są papiery wartościowe. [3]

## **Instytucja finansowa**

Podmiot gospodarczy, który zajmuje się obrotem środków finansowych powierzonych przez klientów (udzielanie kredytów, inwestycje, gromadzenie środków pieniężnych).

## **Oferta inwestycyjna**

Przekazanie wkładu pieniężnego na wybraną pożyczkę przez jednego z pożyczkodawców. Wiąże się to ze wstępną akceptacją oferty pożyczkowej na warunkach wcześniej określonych przez osobę wystawiającą aukcję pożyczkową (*pożyczkobiorcę*).

**Pożyczka**

Przekazanie określonego podmiotowi pieniędzy na określony czas z zastrzeżonym terminem zwrotu.

**Pożyczka społecznościowa (*ang. social lending*)**

Pożyczka wykonywana bezpośrednio pomiędzy osobami fizycznymi poprzez serwisy internetowe, bez pośrednictwa instytucji finansowych.

**Pożyczkobiorca**

Osoba lub instytucja biorąca pożyczkę.[4]

**Pożyczkodawca**

Osoba lub instytucja udzielająca pożyczki.[4]

**RRSO (Rzeczywista roczna stopa oprocentowania)**

Wszystkie koszty realnie ponoszone przez pożyczkobiorcę. Kwotę wyraża się w wartości procentowej całkowitej kwoty kredytu w stosunku rocznym. (Przykładowo, Tomek, pożyczając 1000 zł na 2 lata musi oddać 1200 zł. Roczna opłata za kredyt wynosi  $200/2 = 100$  zł, co jest wartością 10% z całkowitej pożyczonej kwoty. RRSO wynosi wtedy 10%.)

**Saldo**

Stan konta w wybranej walucie lub jednostce aktywów (ilość akcji, kruszcu), po uwzględnieniu wszystkich transakcji rozliczonych i nierozliczonych w danej chwili.

**Windykacja**

Dochodzenie własności za pomocą dostępnych środków prawnych. Dążenie do odzyskania pieniędzy przekazanych podmiotowi przez osobę wszczynającą proces odzyskiwania należności.

**Zobowiązania**

Zmuszenie prawem do świadczenia pieniężnego na rzecz wierzyciela.

# Wstęp

## Pomysł napisania aplikacji o tematyce finansowej

Od 2010 roku interesuję się tematyką oszczędzania pieniędzy i pomnażania oszczędności. Początkowo starałem się szukać bezpiecznego miejsca dla oszczędności w lokatach, które do 2012 roku jeszcze przebijają inflację. Lokaty jednodniowe, potocznie znane jako „antybelki”, były popularnym sposobem na ominięcie podatku od dochodów kapitałowych. Po kilkukrotnej inwestycji w lokaty, zauważyłem, że większy zysk można osiągnąć odkładając kapitał na Giełdzie Papierów Wartościowych. Szybka lektura na temat podstaw inwestowania w papiery wartościowe okazała się być dla mnie lekcją pokory. Jeśli zacznę inwestować bez poświęcenia wielu godzin na naukę funkcjonowania giełdy, stracę wszystkie oszczędności. Inwestowanie w akcje jest obarczone wysokim ryzykiem. Wymaga praktyki, wiedzy i ciągłego obserwacji sytuacji na rynku. Nie byłem skory do zmiany planów życia, by wyciągnąć trochę więcej z moich oszczędności.

Szukając w internecie innych możliwości inwestycji natknąłem się na względnie nowy produkt na polskim rynku. Są nim pożyczki społecznościowe – prosta idea wykorzystująca zestaw technologii znany jako Web 2.0. Zawartość serwisu tworzą użytkownicy. Pożyczając sobie wzajemnie pieniądze, omijają banki i zarabiają pieniądze na klasycznych umowach pożyczkowych. Zalety tego rozwiązania, takie jak zysk z inwestycji wyższy niż na lokatach bankowych oraz bardzo niska krzywa nauki, zachęciły mnie do wkroczenia w świat kredytów. Już ponad 6 lat odkładam nadmiar gotówki z przyzwoitym wynikiem, który po uśrednieniu przekracza maksymalny możliwy zysk z lokaty bankowej.

Gdybym posiadał ciągły dostęp do serwisu, osiągnąłbym dużo wyższy zwrot z inwestycji, ponieważ wiedziałbym, kiedy pojawia się tzw. gorąca oferta (pożyczka na bardzo dobrych warunkach, zazwyczaj jest zamykana w ciągu godziny od założenia aukcji). Jednak nie byłbym w stanie poświęcić całego dnia na przeglądanie wszystkich ofert. **Mogę natomiast napisać aplikację mobilną, która poinformuje mnie, gdy nadarzy się okazja godna odciążenia od innych obowiązków.**

Zdaję sobie sprawę, że również inni inwestorzy często wspominają, jak sprzed nosa uciekały im prawdziwe żyły złota – tylko dlatego że nie mieli ciągłego dostępu do strumienia informacji. Biorąc pod uwagę zainteresowanie wokół problematyki, uważam, że temat mojej pracy inżynierskiej znajduje praktyczne zastosowanie i nie jest tylko sztuką dla sztuki. Uważam go za projekt z potencjałem do dalszego rozwoju. Wybrałem temat, który mnie interesuje, czyli instrumenty finansowe. Są one nie tylko bardzo ciekawym lecz także istotnym zagadnieniem. Jest to szczególnie widoczne, gdy spojrzysz na nie z szerszej perspektywy – np. przez pryzmat skutków społeczno-ekonomicznych, jakie można spowodować przez manipulacje rynkami finansowymi. Jedna osoba, operując sporymi ilościami aktywów, jest w stanie znacząco zmienić gospodarkę całego kraju. Przykładem na to są działania George Sorosa, który na spekulacjach walutowych wyprowadził ponad 1 miliard funtów brytyjskich.[5]

Niniejsza praca skupia się wokół rynku pożyczek finansowych, gdzie występują niewielkie obroty

gotówkowe w przeliczeniu na osobę. Ze względu na swój mikro charakter pożyczki społecznościowe cieszą się moim szczególnym uznaniem. Uważam, że pomagają jednostkom wyjść z problemów ekonomicznych, a w wielu przypadkach urzeczywistniają wizję samodzielnego podnoszenia stopy życia. Zalet pożyczek opartych na społecznościach jest wiele. Przedstawiam je w dalszych rozdziałach, zestawiając razem z wadami i ryzykiem inwestycyjnym.

## Podobne rozwiązania w świecie aplikacji mobilnych

Przeanalizowałem rynek aplikacji mobilnych w sklepach: Google Play, App Store, Windows Store, Amazon AppStore oraz F-Droid. W sklepie Google Play znalazłem jedną aktywnie rozwijaną aplikację wspomagającą pożyczki społecznościowe – Kokoid. Jest to nowa aplikacja (z początku 2016 roku), przeznaczona na system Android. Swoją model biznesowy opiera na wyświetlaniu reklam w trakcie działania.

Oprogramowanie wspomagające inwestycje istnieje głównie w formie aplikacji internetowych. Popularnym narzędziem wspomagającym pożyczkodawców jest [www.zndpd.pl](http://www.zndpd.pl) czyli „Zestaw Narzędzi Pożyczkodawców”. Stanowi ono rozbudowane narzędzie umożliwiające przeglądanie informacji o aukcjach, pożyczkobiorcach i statystykach spłacalności z różnych serwisów. Wszystkie dane zebrane są jednym miejscem w formie strony z raportem. Funkcjonalność portalu zorientowana jest na wyświetlaniu statystyk i generowaniu raportów dotyczących własnych inwestycji.

Nie dostrzegłem żadnych aplikacji o podobnej tematyce (przynajmniej na polskim rynku), na których mógłbym się wzorować. Z tego powodu większość zaimplementowanych funkcjonalności to pomysły nowe, gotowe do przetestowania poza środowiskiem deweloperskim.

## Rozważane formy wdrożenia projektu

Najlepsze rozwiązanie pod względem satysfakcji użytkownika końcowego stanowi stworzenie zestawu narzędzi w formie aplikacji internetowej. Aplikacja ta udostępniałaby API dla aplikacji mobilnych lub aplikacji przeglądarkowej. Główną zaletą tego pomysłu jest dostęp do usługi niezależnie od posiadanej platformy mobilnej. Dodatkowo istnieje możliwość dostosowania danej aplikacji mobilnej do ergonomii wskazanego systemu operacyjnego, co zapewniłoby najwyższą wygodę użytkowania.

Niestety wdrożenie tej idei byłoby zbyt pracochłonne na jednoosobowy projekt. Aby go wdrożyć, należałoby zastosować podział na kilka niezależnych komponentów, wśród których każdy wykorzystywałby inne technologie. Zazwyczaj takie projekty realizowane są we współpracy wielu specjalistów z różnych dziedzin.

Z powyższych względów, najlepszym sposobem na zrealizowanie pomysłu jest napisanie prostej aplikacji mobilnej działającej pod kontrolą systemu Windows Phone 8.1.

Czynniki, które zdecydowały o implementacji projektu w formie aplikacji to: zajęcie z programowania Windows Phone, znajomość platformy .NET, doświadczenie w tworzeniu interfejsów użytkownika z wykorzystaniem XAML'a.

## Jaki jest cel aplikacji?

Zależy mi, by osoba korzystająca z aplikacji mogła w każdej chwili przeglądać najważniejsze informacje o wybranych przez siebie aukcjach, a także dowiadywać się w czasie rzeczywistym o nowych ofertach pożyczkowych. Umożliwi jej to w ciągu kilku godzin podjąć decyzję o nowej inwestycji. Celem zasadniczym jest **skrócenie czasu potrzebnego do podpisania umowy o pożyczkę**.



---

By tego dokonać, zaprezentuję inwestorowi najistotniejsze dane. Przypomnę, gdy zajdzie potrzeba, że pojawiła się kolejna oferta oraz pozwolę wyszukać aukcje, które najbardziej interesują inwestora.

## Założenia projektowe

Cała praca skupia się na ułatwieniu dostępu do korzystania z najpopularniejszego serwisu pożyczek społecznościowych w Polsce. Aplikacja ma być bezpieczna, stabilna, niezawodna. Szybkość działania jest kwestią drugorzędą, gdyż aplikacja operuje na zbyt małych zbiorach danych, by ich przetwarzanie miało powodować znaczące spowolnienia w responsywności. Myślą przewodnią jest dostarczenie wyłuskanych informacji w odpowiednim momencie.

Kopra jest zgodna z wytycznymi Microsoftu dot. projektowania aplikacji mobilnych dla systemu Windows Phone: „*Aplikacja ma robić jedną rzecz i ma ją robić dobrze*”[6] oraz „*Do more*”[7]. Oba hasła odnoszą się do skupienia na produktywności i komforcie użytkownika.

## Korzystanie z serwisów pożyczkowych na smartfonach

Na polskim rynku nie istnieje obecnie żadne rozwiązanie łączące świat mobilny z rynkiem pożyczek społecznościowych. Brak natychmiastowego dostępu do własnych inwestycji wywołuje stres i zniechęca do poważnej aktywności, ponieważ decyzje powinny być podejmowane jak najkrótszym czasie.

Poszukiwanie nowych ofert na przeglądarce telefonu jest bardzo niewygodne. Ponadto korzystanie na małym ekranie ze starych, nieresponsywnych interfejsów serwisów internetowych wymaga od użytkownika wiele cierpliwości. Instytucje finansowe nie wprowadzają ułatwień dostępu dla urządzeń mobilnych.

Jako osoba inwestująca wziąłem sprawy we własne ręce i stworzyłem aplikację mobilną. Teraz, aby uzyskać dostęp do najnowszych okazji pożyczkowych, wystarczy posiadać urządzenie wyposażone w system operacyjny Windows Phone 8.1 lub nowszy oraz połączenie z internetem.

Uważam, że Kopra ułatwi użytkownikom systemu Windows Phone 8.1 szybki dostęp do informacji.



# Rozdział 1.

## O pożyczaniu

### 1.1 Krótka historia pieniądza

Pożyczki istnieją od tysięcy lat. Pierwsze informacje o pożyczaniu pieniędzy wywodzą się ze starożytnej Mezopotamii [8]. Zajęcie było to tak popularne, że grupy osób zaczęły się specjalizować w pośrednictwie między tymi, którzy mieli pieniądze oraz tymi, którzy te pieniądze chcieli pożyczyć. Dzisiaj te grupy stanowią banki. Wypaczając idee pożyczania pieniędzy od osób posiadających pieniądze (pożyczkodawców) do osób biorących te pieniądze na pewien czas (pożyczkobiorców), banki stały się przedsiębiorstwami szukającymi zysków na obrocie pieniędzmi, nie tylko ich pożyczaniu. Doprowadziło to do bardzo wielu niebezpiecznych zachowań. Najgorszym z nich jest system rezerw częściowych. Działa on do czasu, kiedy następuje zalanie rynku toksycznymi pożyczkami, następnie toksyczne pożyczki nie są spłacane i bank nie jest w stanie utrzymać płynności finansowej, bo jak się okazuje większość jego pieniędzy to zabezpieczenia na papierze, których nie da się wyegzekwować. Kolejnym zagrożeniem, jakie niesie ze sobą system rezerw częściowych jest stale zagrożenie wahaniami nastrojów społecznych. W wyniku plotki rozpущzonej wśród klientów banku bądź czasowej niewypłacalności depozytów, może nastąpić zjawisko paniki bankowej. Wielu ekonomistów zgadza się z poglądem, iż dobrze zarządzany system rezerw częściowych prowadzi do szybszego rozwoju gospodarczego i uniezależnienia bogactwo kraju od złóż naturalnych.[9]

### 1.2 Idea pożyczek społecznościowych

Z języku angielskim znane jako „*peer-to-peer lending*” (w skrócie P2P). Zbiór czynności polegających na korelowaniu osób potrzebujących kredytu gotówkowego z ludźmi chętnymi do udzielenia części kredytu przez platformę online.[10] Dla osób z dodatkową gotówką jest to sposób na naukę oceny ryzyka inwestycji niewielkim kosztem (stawki wejściowe do inwestycji są bardzo niskie). Szczególną zachętą jest tutaj relatywnie duży zwrot z inwestycji biorąc pod uwagę współczynnik ryzyka. W Polsce serwisy pożyczek społecznościowych upowszechniły się, kiedy oprocentowanie lokat spadło poniżej 5% (zaraz po zablokowaniu lokat jednodniowych).[11] Rynek pożyczek otworzył się na zastrzyk gotówki w mniej bezpieczne umowy z wyższym oprocentowaniem. Doprowadziło to do sytuacji obniżenia oprocentowania pożyczek kwalifikowanych jako „wysokiego ryzyka”, wyższej niespłacalności przy stosunkowo niskim oprocentowaniu. Sytuacja po roku ustabilizowała się. Ostatecznie wszystkie serwisy w Polsce poprawiły wymogi weryfikacji finansowej by udzielić pożyczki, a oprocentowanie maksymalne zostało obniżone z 25% do 10% (stan na październik 2016). W efekcie te instrumenty stały się niewiele bardziej opłacalne od lokat sprzed ustawą blokującą

omijanie podatku od dochodów kapitałowych.

## 1.3 Jakie są ryzyka związane z tym instrumentem finansowym?

- **Urząd Skarbowy** – zyski z przychodów kapitałowych są opodatkowane 19-procentową stawką (na rok 2016). Jedyną jasną kwestią tutaj jest, iż należy podatek od każdego wypracowanego zysku zapłacić. Problem pojawia się z interpretacją prawa, kiedy należy założyć działalność gospodarczą i w jaki sposób rozliczać się z urzędem skarbowym oraz fiskusem. Naczelny Sąd Administracyjny w sprawie o sygnaturze II FSK 1472/10 orzekł, że regularne inwestycje w pożyczki nosi znamiona działalności gospodarczej.[12] Jednak w sprawie o sygnaturze II FSK 142/10 ten sam sąd wydał wyrok odwrotny, uznając wówczas, że pożyczanie pieniędzy nie nosi znamion prowadzenia działalności gospodarczej. [13] Mając na uwadze powyższe wyroki, przed rozpoczęciem regularnego pożyczania, należy zwrócić się do dyrektora właściwej sobie Izby Skarbowej z zapytaniem o sposób rozliczania. [14]
- **Windykacje.** Jak wygląda odzyskiwanie wierzytelności przy braku spłat? Pierwszą rzeczą, którą należy zrobić to próbować rozmowy z wierzycielem, wyjaśnić przyczyny opóźnień i znaleźć sposób na najszybszy powrót do regularnych spłat. Jeśli kontaktu nie można nawiązać natychmiast, można skorzystać z monitów udostępnianych w serwisach. Są one dostępne bezpośrednio na stronie z ofertą pożyczkową. Wysłanie monitu, jest przypomnieniem w formie SMS/telefonu/wiadomości email o zaległych ratach, monit stanowi dodatkowe obciążenie finansowe dla spłacającego, dlatego rozwiązanie nie jest zalecane jako pierwsza forma "kontaktu". Kolejnym krokiem jest podjęcie działań prawnych, tu zaczynają się problemy dla inwestora, gdyż należy zdecydować czy ściągać samodzielnie, wynajmując własną firmę prawniczą, czy skorzystać z usług dostępnych w serwisie. Jako drobny inwestor, przy kwotach 100–500 zł najoptymalniejszym rozwiązaniem jest wybór ścieżki od kontaktu z PB do monitów. Jeśli nie można dojść swoich roszczeń polubownie, dobrym rozwiązaniem jest wspólna windykacja większości PD aukcji. Inwestorzy wspólnie wynajmują firmę windykacyjną, gdzie za 10-15% odzyskanej kwoty, można liczyć na większe prawdopodobieństwo odzyskania pieniędzy. Jeśli firma nie odzyska długu, nie dostaje wynagrodzenia.

## 1.4 Jakie zyski można osiągnąć?

Średnie oprocentowanie w serwisie Kokos wynosi 16,87% [15] (na dzień 28.10.2016). Stopa lombardowa ustalona przez Narodowy Bank Polski wynosi 2,5% [16] (ustalona w dniu 05.03.2015).

Maksymalne oprocentowanie dla pożyczki w skali roku wynosi stawka stopy lombardowej pomnożona razy cztery. Średnie oprocentowanie w serwisie jest wyższe niż 10%, ponieważ jest liczone od czasu powstania serwisu, kiedy to stawka lombardowa wynosiła 6,25% co dawało maksymalny zysk do 25% rocznie. Stawka oprocentowania rośnie również kiedy pożyczka jest spłacana przed czasem (jest to możliwe, szczególnie, kiedy PB buduje swoją reputację i zobowiązuje się spłacić 3 letnie zobowiązanie w ciągu 1 roku). Z moich osobistych obserwacji wyłania się obraz zwrotu z inwestycji na poziomie 12% rocznie, wliczając w to niespłacone zobowiązania. Kosztem opłacalności, który często nie jest brany pod uwagę jest czas. Do zysków należy wliczyć również czas poświęcony na analizę i pilnowanie inwestycji, kontakt z pożyczkobiorcą i innymi wpłacającymi. Opłacalność (gdzie zysk przekracza zainwestowany czas) można zaobserwować przy kwotach inwestycji przekraczających 500 zł miesięcznie. Utrzymywanie się z odsetek na serwisie wymaga przejścia na etat by

zarządzać narastającą ilością pożyczek (w tym wypadku można zająć się również inwestowaniem w akcje).

## 1.5 Porównanie najpopularniejszych instrumentów finansowych

Zysk rośnie wraz z ryzykiem. Inwestując aktywa, zawsze istnieje ryzyko straty części lub całości inwestycji. Istnieje prawie wykładnicza zależność między ryzykiem utraty inwestycji z wysokością osiągniętych zysków. Poniżej prezentuję prostą tabelę z porównaniem przykładowych zysków i ryzyka dla poszczególnych instrumentów finansowych.

<i>Właściwość</i>	<b>Obligacje</b>	<b>Lokata</b>	<b>Fundusz inwestycyjny</b>	<b>Pożyczka społecznościowa</b>	<b>Giełda</b>
<b>Ryzyko</b>	Brak	Brak	Niskie	Średnie	Wysokie
<b>Rentowność</b>	Znikoma	Niska	Średnia	Średnia, Wysoka	Wysoka
<b>Czas inwestycji</b>	Min. 60 mies.	3–12 mies.	Min. 60 mies.	12–36 mies.	Min. 36 mies.
<b>Poziom skomplikowania</b>	Podstawowy	Podstawowy	Łatwy	Średni	Wysoki

## 1.6 Porównanie polskich serwisów pożyczkowych

Istniejące serwisy w Polsce:

1. Finansowo.pl [17]
2. Zakra.pl [18]
3. Kokos.pl [19]
4. Lendico.pl [20]
5. Zakramini.pl [21]
6. AppleCredit.pl [22]
7. CapitalClub.pl [23]

<i>serwis</i>	<b>Użytkownicy</b>	<b>Udzielone pożyczki</b>	<b>Spłacone zobowiązania</b>	<b>Spłacalność</b>	<b>Średnie oprocentowanie</b>	<b>Charakter</b>
<b>Finansowo</b>	b/d	b/d	85 813 tys. zł	91,9%	5,49%	Chwilówki
<b>Zakra</b>	39 tys.	b/d	b/d	b/d	10%	Konsument
<b>Kokos</b>	309 tys.	126 tys.	144 426 tys. zł	93%	16,86%	Konsument
<b>Zakramini</b>	17 tys.	44 tys.	13 706 tys. zł	b/d	b/d	Chwilówki
<b>AppleCredit</b>	8 tys.	2 tys.	1 135 tys. zł	b/d	9%	Chwilówki
<b>CapitalClub</b>	1 tys.	0,1 tys.	4 000 tys.	b/d	b/d	Inwestycje

W porównaniu brakuje serwisu Lendico. Ten niemiecki serwis zawiesił działalność inwestycyjną, od kilku miesięcy nie ma ofert inwestycyjnych. Prawdopodobnie ze względu na niskie zainteresowanie zostanie wkrótce wycofany z polskiego rynku.

### 1.7 Wybór systemu do implementacji

Pierwszym i obecnie najpopularniejszym serwisem jest Kokos. W ciągu 8 lat działalności pożyczono 143,4mln zł na 126 738 aukcjach. Dla porównania CapitalClub informuje na stronie głównej o sumie 4mln zł przez 102 udzielone pożyczki. Kokos, którego właścicielem jest firma Bluemedia, posiada największą bazę aktywnych inwestorów, wiele innowacyjnych zabezpieczeń jak Friendly-Score, responsywny interfejs, przelewy ekspresowe, raporty inwestora, podstawowe API do zapytań o aukcję. Ważnym aspektem jest również dostęp do interfejsu programistycznego udostępnionego dla programistów. Ze wszystkich platform, API jest udostępnione na trzech – Kokos, Zakra, Zakramini. Zakra i Zakramini pozwalają korzystać programiście/inwestorowi z zapytań do serwisu po osobistej prośbie o wygenerowanie klucza. Stanowi to wielką przeszkodę, każdy kto chce sprawdzić działanie funkcjonalności jest zobowiązany wystosować pisemną prośbę o wygenerowanie klucza i po kilku dniach otrzyma dostęp do WebAPI. Kolejnym mankamentem jest bardzo okrojony zakres zwracanych informacji. Dostępne są następujące wywołania (wraz z odpowiedziami):

#### **getUserData(id użytkownika)**

[id użytkownika, nazwa użytkownika, rok urodzenia, miasto zameldowania, id użytkownika w serwisie Zakramini, nazwa użytkownika w serwisie Zakramini, kwota zadłużenia w serwisie Zakramini, czas opóźnień spłaty na w serwisie Zakramini]

#### **getAuctionData(id aukcji)**

[id, id pożyczkobiorcy, oprocentowanie, kwota pożyczki, numer aukcji, bonus pożyczkodawcy, status aukcji, data utworzenia]

#### **getRunningLoansFromUser(id użytkownika)**

[tablica z aukcjami jak getAuctionData]

#### **getAuctionInstallmentsData(id aukcji)**

[id aukcji, dzień płatności, data pierwszej spłaty, ilość spłaconych rat, ilość zaległych rat]

Zwracają one tylko najbardziej podstawowe informacje [24], właściwie szczytkowe. Nie można z nich złożyć porządnej informacji kredytowej.

Pozostaje jeszcze Kokos ze swoim interfejsem. Pozwala na uzyskanie znacznie bardziej złożonych informacji o aukcji, użytkownikach, stanie finansowym. Wygenerowanie klucza jest też niesamowicie proste, wystarczy kliknięcie w przycisk „Wygeneruj klucz” i gotowe, na rok klucz jest aktywny. Mam nadzieję, że po przedstawieniu powyższych informacji jest jasne, że wyboru, który serwis obsłużyć, nie istnieje.

### 1.8 WebAPI

Interfejs programistyczny wchodzi w skład Narzędzi użytkowników. Dokumentacja [25] jest kompletna (parę brakujących elementów można szybko wywnioskować z kontekstu). Aby rozpocząć korzystanie, należy wygenerować własny klucz, zapoznać się z regulaminem korzystania oraz dokumentacją użytkownika. Dokumentacja ma formę listy funkcji dostępnych dla programisty. Wybierając konkretną metodę, otrzymuje się stronę z opisem argumentów (parametrów) funkcji, listę zwracanych elementów, strukturę, przykład użycia, przykładowe zapytanie i odpowiedź w formie

XML. Odpowiedź na zapytania można dostać w formacie XML, JSON, HTML. Zapytania wykonuje się w modelu REST, wszystkie parametry są przekazywane przy użyciu GET w postaci:

`https://kokos.pl/webapi/search?key=<klucz>&<parametry>`

Użytkownik może wykonywać jedno zapytanie na sekundę. Przy częstszych odpytaniach serwis zgłosi błąd 503 „*Service Temporarily Unavailable*”. Ta sama sytuacja pojawi się, kiedy stronę internetową odświeża się w przeglądarce zbyt często. Do dokumentacji dołączony jest również słownik, bez którego nie mógłbym złożyć odpowiedniego wywołania. Część słownika jest zaszyta w aplikacji, nie można dynamicznie go pobierać i aktualizować kontrolek w aplikacji. Jednak są to elementy niezmiennie, takie jak województwa, statusy spłaty, bądź status aktywności pożyczki. Z istotnych informacji trzeba jeszcze pamiętać o wygenerowaniu klucza co roku. Aplikacje korzystające z klucza muszą go odnowić. Data ważności klucza jest wyświetlana na stronie dokumentacji w zakładce „*Klucz WebAPI*”. Niestety próba odświeżenia klucza przed jego wygaśnięciem nic nie zmieni, należy odczekać do jego wygaśnięcia, by aktywować go ponownie na rok. Same odpowiedzi z serwera bywają nieprzewidywalne. Podczas tworzenia aplikacji na zapytanie o listę inwestorów, raz otrzymałem listę w formacie JSON. Kiedy inwestorów nie było, zamiast pustej listy otrzymywałem jeden pusty element. W okolicy 10 dnia każdego miesiąca, wieczorem należy uzbroić się w cierpliwość, kiedy inwestorzy szturmują serwis w poszukiwaniu najlepszych aukcji. Przez opóźnienia serwer wysyła częściej błąd 503 jako odpowiedź.





## Rozdział 2.

# Wykorzystane technologie i narzędzia

W tym rozdziale przedstawiam technologie i narzędzia, które w znaczący sposób pomogły zrealizować projekt. Skupiam się na pobieżnym opisie technologii, prezentuję również sposób implementacji komponentów aplikacji.

### 2.1 Microsoft Visual Studio Community 2015

Miejsce, w którym spędziłem najwięcej czasu, napisałem i złożyłem projekt jest Visual Studio (VS) w wersji Community Edition 2015. Jest to obecnie najbardziej podstawowa ze wszystkich wersji IDE (Integrated Developer Environment) przeznaczonych do tworzenia aplikacji mobilnych na telefony z systemem Windows Phone 8.1. Narzędzia dostępne w tym oprogramowaniu są w zupełności wystarczające by tworzyć aplikacje typu Kopra. Należy pamiętać, że VS jest środowiskiem wymagającym sporych zasobów sprzętowych. Na stronie producenta [26] można znaleźć informacje o minimalnych wymaganiach: procesor 1.6 GHz lub szybszy, 1Gb pamięci RAM, 4Gb miejsca na dysku twardym, karta graficzna wspierająca DirectX 9.0c i rozdzielczość przynajmniej 1024 x 786. Aby programować urządzenia Windows Phone należy mieć wielokrotnie mocniejszy sprzęt, szczególnie jeśli programuje się przy użyciu emulatorów.

### 2.2 Język C# 5.0

Język programowania Microsoft Visual C# został zaprojektowany z myślą o tworzeniu aplikacji działających z zestawem narzędzi .NET Framework. Firma Microsoft projektowała go jako połączony, a zarazem prosty język orientowany obiektowo z bezpiecznym typowaniem. [27] Składnia wywodzi się z rodziny języków C. Dziedziczy wiele najlepszych cech języka C++ i Microsoft Visual Basic, odrzucając większość ich anachronizmów, tworząc bardziej logiczną strukturę języka. Pierwsza wersja oficjalnie zadebiutowała w 2001 roku. Wielu programistów uważało go za próbę skopiowania Javy, jednak kolejne wersje przewyższają prostotą składni i klarownością instrukcji swój domniemany pierwowzór. Premiera C# 2.0 ze środowiskiem Visual Studio 2005 wprowadzała kilka nowych elementów do języka, takich jak typy generyczne, iteratory, metody anonimowe. Uważam, że kolejna wersja języka, C# 3.0 przyczyniła się do największej adopcji wśród programistów. Wszystko przez innowacyjne konstrukcje, które konkurencja wprowadzała przez kilka następnych lat. Moją ulubioną nowością tego wydania jest LINQ (Language-Integrated Query). Również wersja

C# 5.0 daje wrażenie dynamicznego rozwoju języka. Projektanci ułatwili tworzenie wywołań asynchronicznych przez dodanie modyfikatorów metod `async/await`. Kolejne wersje języka wprowadzają udogodnienia z ułatwiające największe wyzwania stojące przed programistą. Każda wersja języka jest ściśle powiązana ze środowiskiem Visual Studio C# oraz zestawem narzędzi .NET Framework, choć numeracja wersji nie zawsze się pokrywa.

## 2.3 .NET Framework

W odniesieniu do zmian wprowadzonych przez Microsoft w 2016 roku w związku z wprowadzeniem nowego podziału architektury .NET, ograniczam się do Standardowej Biblioteki .NET (ang. .NET Standard Library) oraz do .NET Framework ograniczony do podzbioru WPF.[28] .NET Framework jest platformą programistyczną wydaną w 2002 roku przez Microsoft. Do niedawna była to jedyna platforma dla programistów .NET. W platformie znajduje się zestaw klas biblioteki oraz API systemu Windows, które znacznie podnoszą produktywność programowania aplikacji typu desktop. Kolejne wydania platformy .NET są ściśle powiązane z nowymi wydaniem języka C#. Wersja Frameworka używana przez Koprę jest oznaczona numerem 4.5.1 i jest kompatybilna z Visual Studio 2013 na Windows 8.1 wraz z wersjami późniejszymi.

## 2.4 Windows Phone 8.1 SDK

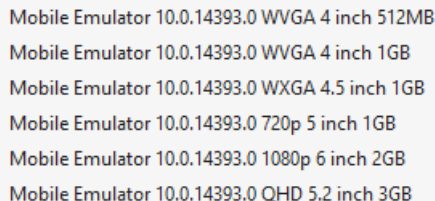
Microsoft ukrył stronę z Windows Phone 8.1 SDK oraz zestawem emulatorów na rzecz promocji Windows 10 Mobile. Zestaw narzędzi jest dość trudny w odnalezieniu. By się dostać do zasobu, musiałem użyć zewnętrznej wyszukiwarki, ponieważ ta ze strony Microsoft.com nie zwraca wyników. Żeby zdobyć zestaw narzędzi programistycznych na tą platformę, trzeba przeszukiwać archiwa Microsoftu. [29] Do programowania na system Windows Phone potrzebne jest urządzenie fizyczne (preferowana Nokia Lumia) bądź emulator, do którego wymagany jest procesor ze wsparciem SLAT (Second Level Address Translation). Należy również pamiętać, że do komfortowej pracy z emulatorem potrzeba przynajmniej 8Gb pamięci RAM. Zestaw SDK zawiera 6 różnych konfiguracji emulatorów:



Rysunek 2.1: Zestaw konfiguracji emulatorów dostępny w Windows Phone 8.1 SDK

Aplikację przetestowałem na każdym z wyżej wymienionych urządzeń wirtualnych, w orientacji pionowej oraz na urządzeniach Nokia Lumia 925, Microsoft Lumia 530, Nokia Lumia 625.

Kopra została również uruchomiona pod emulator systemu Windows Mobile 10 i działa poprawnie pod poniższymi konfiguracjami. Nie została przetestowana dokładnie na nich, również nie miałem dostępu do urządzeń fizycznych z systemem WM 10, stąd użytkownik powinien używać oprogramowania na tym systemie na własną odpowiedzialność.



Mobile Emulator 10.0.14393.0 WVGA 4 inch 512MB
Mobile Emulator 10.0.14393.0 WVGA 4 inch 1GB
Mobile Emulator 10.0.14393.0 WXGA 4.5 inch 1GB
Mobile Emulator 10.0.14393.0 720p 5 inch 1GB
Mobile Emulator 10.0.14393.0 1080p 6 inch 2GB
Mobile Emulator 10.0.14393.0 QHD 5.2 inch 3GB

Rysunek 2.2: Zestaw konfiguracji emulatorów dostępny w Windows Mobile 10 SDK

## 2.5 Newtonsoft.Json

Najpopularniejsze narzędzie do pracy z danymi w formacie JSON. Oprogramowanie działa na jednej z najbardziej liberalnych licencji, MIT. [30] **Wśród bibliotek dostępnych w menadżerze pakietów NuGet, zajmuje pierwsze miejsce mając ponad 15 milionów pobrań.** [15] Json.NET znacznie uprościł proces przetwarzania danych otrzymanych z serwisu. Zgodnie z opisem na stronie, biblioteka jest bardzo łatwa w użyciu, nawet przy złożonych typach danych. Znalazłem się jednak kilkukrotnie w sytuacji, gdzie musiałem rozbudować mechanizm serializujący o dodatkowe warunki. Działo się tak kiedy serwis zwracał mi różne rodzaje danych dla jednego zapytania. Dla pobierania elementów ze struktury HTML, których nie można pobrać przez API serwisu, istnieje mechanizm wspierający XPath. JSONPath [31] jest wbudowany w bibliotekę i świetnie wspiera pobieranie elementów wg. ścieżki i natychmiastowe ich serializowanie do JSON'a.

## 2.6 Git i GitHub

W utrzymaniu ładu z kolejnymi wersjami pomógł mi Git. Jest to rozproszony system kontroli wersji stworzony przez Linusa Torvaldsa jako narzędzie do zarządzania kodem źródłowym Linuksa, kiedy relacje z firmą BitMover pogorszyły się i projekt Linux stracił możliwość korzystania z BitKeeper'a za darmo. Git został zaprojektowany w oparciu o 5 cech, które do dziś stanowią jego podstawę: szybkość, prosta konstrukcja, wsparcie dla równoległego rozwoju wielu wersji jednocześnie, w pełni rozproszony, z możliwością zarządzania wielkimi projektami. [1] Przekonującym argumentem do użycia właśnie tego systemu kontroli wersji jest dostęp do serwisów zintegrowanych z oprogramowaniem. Zmiany przechowywane są lokalnie w formie lokalnej bazy. Większość akcji wykonywanych na repozytorium git, dodaje dane do bazy. To sprawia, że popsucie czegoś bez możliwości przywrócenia stanu jest prawie niemożliwe. Dlatego można śmiało psuć projekt, bez obaw na utratę danych. Tworząc większe funkcjonalności w projekcie korzystałem z możliwości tworzenia własnych gałęzi (branch), a następnie po zaimplementowaniu łączyłem je do głównej linii produkcyjnej (master).

Zamiast korzystać z własnego serwera systemu kontroli wersji, użyłem darmowego i najbardziej popularnego serwisu oferującego usługi przechowania repozytorium założonego w Git. Na chwilę obecną jest największą platformą hostującą projekty open-source. Ponad 18 milionów użytkowników i 48 milionów założonych projektów daje mi gwarancję, że w trakcie tworzenia projektu serwis nie zostanie wyłączony z użycia. Usługa oferuje możliwość przechowania kodu widocznego dla wszystkich (Public) oraz tylko dla wybranych użytkowników (Private). GitHub jest darmowy dla pierwszego typu hostingu. [32] Kod źródłowy jest widoczny dla każdego, jeśli ktoś zechce wykonać własną gałąź aplikacji i w jakikolwiek sposób ją poprawić, będzie to dla mnie miłym zaskoczeniem.

### 2.7 LinqPad

Narzędzie wykorzystałem do szybkiego modelowania kodu metod. Jest to produkt powszechnie używany do modelowania fragmentów kodów w środowisku .NET. Program jest używany do pisania interaktywnych zapytań SQL z wykorzystaniem LINQ oraz pisania kodu C# bez konieczności uruchamiania Visual Studio. [33] Produkt jest tworzony na modelu Freemium, co oznacza, że program jest darmowy z wyłączeniem pewnych funkcjonalności, za które należy zapłacić. [34]

### 2.8 ILSpy

Przeglądarka plików assembly i dekompilekator kodu .NET, oparty o licencję MIT. Projekt hostowany jest na GitHub, gdzie można pobrać jego źródła. [35] Narzędzie początkowo używałem do dekompilacji biblioteki HtmlAgilityPack, która przez zbyt wiele błędów zmusiła mnie do przeglądania jej wewnętrznych metod. Na późniejszym etapie ILSpy pomógł mi zrozumieć dokładniej część biblioteki .NET zawartej w przestrzeni nazw System.Net.Http. ILSpy stał się szczególnie popularny, kiedy narzędzie Microsoftu o nazwie IL DASM, przestało być częścią pakietu Visual Studio. Przed wersją Visual Studio 2012, IL DASM był dostępny z poziomu folderu Microsoft SDK Tools jako narzędzie do zestawu razem z IL ASM. Ten zaś służy jako narzędzie do generowania kodu wykonywalnego z kodu pośredniego. Obecnie te narzędzia dostępne są w formie konsolowych programów. ILSpy jest w pełni graficznym narzędziem. Uważam, że jest najlepszym darmowym narzędziem do przeglądania plików assembly.

## Rozdział 3.

# Budowa projektu

Projekt został zbudowany w oparciu o bibliotekę Windows Phone SDK z wykorzystaniem wzorca architektonicznego MVVM. Z narzędzi zewnętrznych, zapewniłem integrację z Web API serwisu Kokos.

### 3.1 Wzorzec MVVM

W tym rozdziale przedstawię wzorzec architektoniczny MVVM, jego zalety i wady oraz sposób w jaki zaimplementowałem go w projekcie.

#### 3.1.1 Opis wzorca Model View ViewModel

Wprowadzony przez Microsoft w 2005 roku, wzorzec architektoniczny, opiera się na tych samych założeniach co zaprezentowany lata wcześniej model prezentacji (Presentation Model). PM dziedziczy po Model Widok Prezenter MVP - (*ang. Model View Presenter*), wzorzec architektoniczny z początku lat 90 ubiegłego wieku. Z tym, że wzorzec PM jest szczególnie przydatny w programowaniu złożonych interfejsów użytkownika. Na systemie Windows, Presentation Model bardzo dobrze współpracuje z interfejsem użytkownika zbudowanym z biblioteką Windows Presentation Foundation (WPF) oraz biblioteką Silverlight. Właśnie dlatego Microsoft zbudował wersję wzorca PM specjalnie pod WPF i nazwał go Model-View-ViewModel (MVVM). [2] MVVM używa wielu specyficznych możliwości i narzędzi znanych z platformy Silverlight takich jak wiązanie danych. Wprowadzenie wzorca MVVM w aplikacjach opierających się na technologiach WPF, Silverlight jest zalecanym podejściem, ponieważ są to bardzo ułatwiają jego implementację.

Do zalet Model View ViewModel względem tradycyjnego podejścia (przeciągania kotrolek z Toolboxa i pisanie dalej w formie Code-behind) są:

- Umożliwia oddzielenie logiki od sposobu wyświetlania. W dłuższej perspektywie ułatwia to rozwój, testowanie i utrzymanie kodu.
- Jest wspierany przez platformy XAML.
- Kod interfejsu jest niezależny od platformy kodu wykonywalnego.
- Umożliwia rozdział między projektanta interfejsu i programistę, tzn. każdy z nich może pracować oddzielnie nad swoją odpowiedzialnością.

Wzorzec podzielony jest na trzy części, które wspierają testowanie aplikacji przez rozdzielanie elementów interfejsu użytkownika od logiki aplikacji. Taki rozdział pozwala na wymianę komponentów aplikacji, wewnętrzną zmianę implementację, wyizolowane testy jednostkowe. Komponenty na najwyższym poziomie abstrakcji nie wiedzą o sobie zbyt wiele. Widok (V) wie o istnieniu **widok-modelu** (*ang. ViewModel*) (VM), widok-model (VM) wie o istnieniu modelu, jednak model nie wie o widok-modelu (VM) i widok-model (VM) nie wie nic o Widoku (V).

**Widok (*View*)** odpowiada za strukturę, rozkład i wygląd tego co użytkownik otrzymuje na ekranie. W większości przypadków aplikacji Windows Phone, widok zdefiniowany jest jako Strona zaimplementowana przy użyciu XAML'a. Widok ma kilka możliwości na wykonanie kodu dla użytkownika, najczęściej rekomendowanym sposobem jest użycie właściwości *Command*.

**Model (*Model*)** jest odpowiedzialny za przechowywanie danych w obrębie swojej domeny oraz logiki biznesowej.

**Widok-model (*ViewModel*)** rozdziela logikę biznesową od widoku. Zadaniem tego komponentu jest pobranie danych z Modelu i przedstawienie go w formie łatwo przyswajalnej dla widoku. Widok-model również może odpowiadać za stany widoku, jak na przykład przekazanie widokowi stanu ładowania.

#### 3.1.2 MVVM vs MVC

Praca nie byłaby kompletna, gdybym nie wspomniał nic o wzorcu, który wpłynął w znaczący sposób na rozwój MVVM. Wzorzec bardzo podobny, który powstał jednak dużo wcześniej to Model-View-Controller (MVC). Jest to wzorzec architektoniczny, używany do implementacji interfejsu użytkownika. Polega na podziale oprogramowania na trzy niezależne od siebie części. Wzorzec został zaprezentowany po raz pierwszy przez Trygve Reenskaug jeszcze w latach 70tych. [36] Wzorzec dzieli modelowanie domeny, prezentacji i akcji na trzy oddzielne klasy. Model zarządza zachowaniem i danymi w domenie aplikacji, odpowiada na żądania informacji o swoim stanie oraz odpowiada na żądania zmiany stanu, które otrzyma od kontrolera. View stanowi o graficznej części aplikacji, otrzymuje od kontrolera informacje w jaki sposób interpretować dane wejściowe otrzymane od użytkownika. Controller służy do przetwarzania informacji otrzymywanych od użytkownika np. z urządzeń wejściowych. MVC stał się bardzo popularnym wzorcem przy projektowaniu aplikacji typu desktop, a później także Web.

#### 3.1.3 Implementacja MVVM w aplikacji

Aplikacje korzystające z wzorca MVVM mają z góry ustalony szkielet folderów.

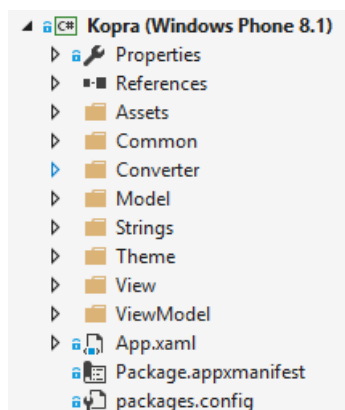
Mój projekt ściśle trzyma się tej struktury wraz z dodatkowymi elementami. Przyjąłem następującą konwencję nazewnictwa, dla każdego widoku dołączyłem sufix „Page”, natomiast każdy widok-model w nazwie ma dołączony sufix „ViewModel”. Każdy widok ma odpowiadający mu jeden widok-model.

```
<Page.DataContext>
    <viewModel:NameViewModel></viewModel:NameViewModel>
</Page.DataContext>
```

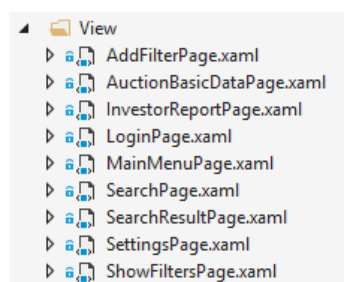
Listing 3.1: Podpięcie widok-modelu do Widoku w pliku XAML

```
private AddFilterViewModel _viewModel;

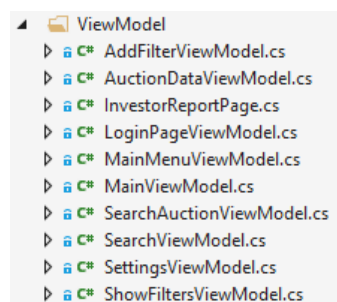
public konstruktor() {
```



Rysunek 3.1: Struktura folderów w projekcie aplikacji Kopra



Rysunek 3.2: Lista plików widoków w projekcie



Rysunek 3.3: Lista plików widok-modeli w projekcie

```
        _viewModel = DataContext as AddFilterViewModel;  
    }
```

Listing 3.2: Podpięcie widok-modelu w pliku Code-Behind. W konstruktorze inicjalizuje obiekt podając widok-model z którym jest spięty.

Do wiązania danych zaimplementowałem interfejs `INotifyPropertyChanged`.

```
public class MainViewModel : INotifyPropertyChanged  
{  
    public event PropertyChangedEventHandler PropertyChanged;  
  
    public void NotifyPropertyChanged(string propertyName)  
    {  
        PropertyChanged?.Invoke(this,  
            new PropertyChangedEventArgs(propertyName));  
    }  
}
```

Listing 3.3: Implementacja interfejsu `INotifyPropertyChanged` w jednej z klas

Mając tak zaimplementowaną klasę, pozostałe widok-modele dziedziczą po `MainViewModel` używając wywołania metody `NotifyPropertyChanged()` do powiadomienia o zmianie wartości właściwości.

```
public string ApiKeyValue  
{  
    get { return _apiKeyValue; }  
    set  
    {  
        _apiKeyValue = value;  
        NotifyPropertyChanged(nameof(ApiKeyValue));  
    }  
}
```

Listing 3.4: Przykładowa właściwość wykorzystująca mechanizm `NotifyPropertyChanged`

## 3.2 Kokos WebAPI

Rozdział przeznaczony na prezentację interfejsu programistycznego udostępnionego przez zespół BlueMedia. Przedstawiam również niedogodności związane z jego wykorzystaniem, metodę konsumpcji przez Koprę i sposoby poprawienia komfortu użytkowania. Przez rozpoczęciem pracy jedynym wymogiem (oprócz aktywnego konta) jest wygenerowanie klucza, który służy pozwala odróżnić kto wykonuje zapytanie.

### 3.2.1 Opis interfejsu webowego

Zapytania do serwisu wykonuje się przez złożenie zapytania RESTowego. Czyli

`https://kokos.pl/webapi/FUNKCJA?key=KLUCZ&type=FORMAT_DANYCH&PARAMETR=WARTOŚĆ`

Aby uniknąć nadmiernego obciążenia i wykorzystania klucza przez więcej niż jednego użytkownika liczba zapytań jest ograniczona do jednego na sekundę. Przy próbie przekroczenia, zostanie zwrócony w odpowiedzi kod 503. Lista dostępnych funkcji:



**get-auction-data**

Pobranie szczegółowych danych jednej wybranej aukcji.

**get-auctions-by-status**

Pobranie listy pożyczek według statusu.

**search**

Wyszukiwanie aukcji według podanych parametrów.

**get-most-popular-auctions**

Pobranie najbardziej popularnych aukcji.

**get-recent-auctions**

Pobranie ostatnio założonych aukcji.

**get-recent-payments**

Pobranie ostatnich wpłat za raty.

**get-recent-investments**

Pobranie najnowszych inwestycji.

**get-ended-auctions-amount**

Pobranie sumy wartości aukcji zakończonych sukcesem.

**get-service-stats**

Pobranie statystyk serwisu.

**get-user-id-by-nick**

Pobranie ID użytkownika na podstawie jego nick-u.

**get-payment-stats**

Pobranie statystyk spłacalności dla całego serwisu.

**get-vindication-stats**

Pobranie statystyk spłacalności.

Każda z funkcji ma określony spory zakres parametrów oraz tabelę ze zwracanymi wartościami.

### 3.2.2 Implementacja API w aplikacji

W celu uniknięcia powielania kodu podczas zapytań, napisałem klasę, która generuje zapytania do serwisu. Wykorzystuje mniej niż 50% funkcji, dlatego nie widziałem powodu by tworzyć osobny projekt na komunikację z serwisem. Elementy stałe, przechowuję w formie prywatnych elementów klasy.

```
private const string BaseAddress =  
    "https://kokos.pl/webapi/";  
private const string DataType = "&type=json";  
private const string Search = "search?";  
private const string RecentAuctions =  
    "get-recent-auctions?";  
private const string AuctionData =  
    "get-auction-data?";  
private const string Comments = "comments=1";
```

```
private const string Records = "records=";  
private const string Key = "key=";  
private const string Id = "id=";  
private const string Type = "type=";
```

Listing 3.5: Składowe budujące zapytanie do serwisu

Tworzenie parametrów zapytań API schowane jest w klasie RequestGenerator. Przechowuje tam zbiór funkcji do tworzenia całościowych obiektów Uri, gotowych do przekazania użycia przez serwis.

```
public Uri FilteredAuction(string filter)  
public Uri ComposeSearchAuctionQuery(Dictionary<string, string>  
    search)  
public Uri MostRecentAuctions()  
public Uri GetAuctionData(GetAuctionDataParameters parameters)
```

Listing 3.6: Nagłówki metod pozwalających składać zapytania do serwisu.

Komunikacja z serwisem odbywa się przy pomocy klasy KokosConnectionManager. Do komunikacji internetowej, wykorzystana jest klasa HttpClient.

## 3.3 Notyfikacje przez BackgroundTask

### 3.3.1 Opis

Windows Phone 8.1 pozwala wykonywać aplikacji zadania bez konieczności trzymania jej na pierwszym tle. Funkcjonalność ta nazywa się *Background Task* i użyłem jej do powiadamiania użytkownika o nowych aukcjach w serwisie.

### 3.3.2 Implementacja NewAuctionNotifier

*Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur non lectus pulvinar, bibendum est eu, venenatis odio. In faucibus orci vel tortor pellentesque molestie. Nullam vel risus mattis, volutpat arcu non, cursus est. Phasellus facilisis risus id justo consectetur porta. Integer neque lorem, tincidunt in mi id, eleifend consequat quam. Sed lacinia feugiat neque. Sed id eleifend lectus, et vulputate enim. Aenean viverra, diam vitae sodales auctor, risus risus varius metus, vitae bibendum metus nibh et ipsum. Aenean id placerat neque.*

## Rozdział 4.

# Prezentacja interfejsu aplikacji

### 4.1 Etymologia nazwy

Nazwa aplikacji powstała już w trakcie tworzenia, kiedy wiedziałem już z którym serwisem aplikacja będzie współpracować. Szukałem czegoś co jednym słowem opisze sposób myślenia inwestora. „*Jak wyciągnąć największe zyski z inwestycji?*”. W połączeniu z nazwą serwisu, który obsługuje, nie mogło być lepszej nazwy jak Kopra1. Co miało podkreślić, wyciągnięcie esencji z Kokosa.

### 4.2 Korzyści względem wersji przeglądarkowej

Ze względu na brak dobrej przeglądarki internetowej w urządzeniach Windows Phone, stronę trudno jest obsługiwać z poziomu telefonu. Sam serwis nie jest dostosowany do małych ekranów urządzeń mobilnych (brak poprawnie zaimplementowanej responsywności). Kopra została zaprojektowana wyłącznie na takie urządzenia i jest to największa jej zaleta w stosunku do strony internetowej.

### 4.3 Koszt użytkowania aplikacji

Aplikacja jest darmowa, również wersja, która zostanie wystawiona do sklepu będzie udostępniona bezpłatnie, bez reklam. To jest największa różnica względem konkurencyjnego oprogramowania Kokoid, które na każdym ekranie wyświetla użytkownikowi baner reklamowy. Kod źródłowy aplikacji jest dostępny w oparciu o liberalną licencję MIT.

### 4.4 Bezpieczeństwo danych przechowywanych w aplikacji

Aplikacja ma po zalogowaniu ma dostęp do wszystkich danych osobowych umieszczonych w serwisie. Do tych danych należą między innymi: numer i seria dowodu osobistego, adres użytkownika, adresy IP z datą logowania do serwisu, historia kredytowa, aktualne inwestycje, saldo użytkownika, hasło oraz email. Z całego ogromu informacji, w aplikacji trzymane są tylko dane logowania i klucz dostępu do WebAPI użytkownika. Aby poprawić bezpieczeństwo, cały ruch między aplikacją a serwisem Kokos jest przesyłany protokołem SSL. Również aby zapobiec ewentualnym wyciekom danych, aplikacja nie wysyła danych logowania na serwery zewnętrzne.

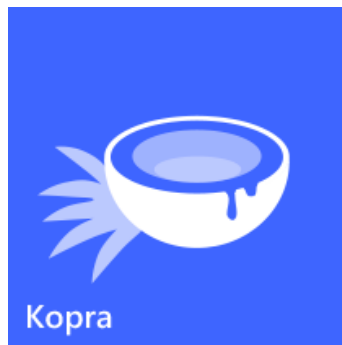
### 4.5 Jak wygląda Kopra?

#### 4.5.1 Logo

Logo aplikacji jest rysunkiem połowy owocu kokosowego z liśćmi drzewa kokosowego. Jak wspomniałem w pierwszych rozdziałach pracy, jest to nawiązanie nazwy do wykorzystywanego serwisu. Logo zostało wykonane w bieli, bez tła, dlatego jest przystosowane do każdego motywu kolorystycznego z nasyconymi barwami (różnymi od bieli).



Rysunek 4.1: Logo na szerokim kafelku



Rysunek 4.2: Logo na kwadratowym kafelku

#### 4.5.2 Splash screen

W trakcie uruchamiania aplikacji, użytkownik może zobaczyć duże logo aplikacji wraz z jej nazwą na niebieskim tle. [miejsce na rysunek](#)

#### 4.5.3 Strona logowania

Pierwszy ekran jaki widzi użytkownik, jest on wykonany w motywie pośrednim. Pozwala na łagodne przejście między domyślnym ciemnym motywem Windows Phone na motyw jasny, który będzie widoczny po zalogowaniu się do serwisu. Na tym ekranie użytkownik jest zmuszony do przekazania loginu (adresu email) oraz hasła bezpośrednio do strony Kokos.pl. Po wybraniu przycisku zaloguj, jest on zastąpiony przez napis „Logowanie...” oraz animację paska postępu. Dodatkowym elementem są komunikaty informujące użytkownika np. o braku połączenia z Internetem. [miejsce na rysunek](#)

#### 4.5.4 Strona główna

Ekran dostępny po zalogowaniu składa się z trzech części. Na górze ekranu po lewej stronie widnieje mała wersja logo oraz nazwy aplikacji. Po prawej stronie, znajduje się nazwa zalogowanego użytkownika. Drugą część stanowią przyciski, symulujące wyglądem kafelki znane w ekosystemie Windows Phone 8. Jest to miejsce agregujące najważniejsze funkcje dostępne dla użytkownika.



Rysunek 4.3: Logo na szerokim kafelku

## 4.6 Instrukcja obsługi do aplikacji

Aby korzystać z aplikacji, należy mieć aktywne konto w serwisie kokos.pl oraz aktywować klucz API pierwszy raz przez przeglądarkę. Po uruchomieniu Kopry, należy podać adres email oraz hasło logowania do serwisu. Po poprawnej weryfikacji, użytkownik zostanie przeniesiony do ekranu głównego, skąd może się dostać do wszystkich najważniejszych funkcjonalności aplikacji. Użytkownik może przeskoczyć bezpośrednio do jednej z trzech najnowszych aukcji wyświetlanych na dole ekranu.

Aby uruchomić zadania w tle, należy mieć dodany przynajmniej jeden filtr w aplikacji, a następnie z ekranu ustawień wybrać z listy, jakie aukcje mają być wyszukiwane w tle. Maksymalnie co 30 minut można otrzymać powiadomienie, z nową aukcją.



## Rozdział 5.

# Podsumowanie

Aplikacja ma w założeniu rozwiązać problem, jakim jest szybki dostęp do oferty pożyczkowej na urządzeniach mobilnych. Skupia się na obsłudze najpopularniejszego serwisu, oferującego usługi „p2p lending”. Kopra ma wyciągnąć najważniejsze informacje, potrzebne do dalszego procesu myślowego czy warto zainteresować się ofertą.

Wymuszenie jasnego motywu, miało pomóc w zdobyciu zaufania użytkownika oraz nadać czystą strukturę interfejsowi. Aby lepiej poznać nawyki użytkowników systemu Windows Phone, zdecydowałem się na użytkowanie systemu przez rok, równoległe z procesem pisania pracy. Jest to główny powód zmiany interfejsu po pierwszej iteracji, gdzie ekrany przypominały zmniejszoną wersję przeglądarkową serwisu. Ze względu na ogrom danych dostępnych użytkownikowi w serwisie, zdecydowałem się na zaprezentowanie szczegółów aukcji na ekranie typu pivot, który pozwala na łatwe grupowanie informacji powiązanych ze sobą tematycznie. Oprogramowanie stworzyłem z myślą o telefonach Windows Phone 8.1 wraz z kolejnymi wersjami, ponieważ jest to system stworzony z myślą o produktywności, szczególnie pod kątem ludzi biznesu.

Na chwilę obecną aplikacja ma możliwość sprawdzania w tle jednego filtru użytkownika. Funkcjonalność można rozszerzyć o wybór ilości powiadomień na godzinę, które użytkownik jest w stanie zaakceptować, dodać listę filtrów które miałyby być sprawdzane i ustawić dla nich priorytety. Również widzę w kolejnych wersjach możliwość tworzenia kopii zapasowej filtrów na dysku OneDrive oraz dzielenie się nimi z innymi inwestorami. Kolejnym kamieniem milowym w rozwoju aplikacji może być automat inwestycyjny, który korzystając z funkcjonalności wyszukiwania aukcji w tle, inwestowałby automatycznie określoną kwotą w znalezione oferty inwestycyjne.

Stworzona przeze mnie aplikacja jest pierwsza w swojej kategorii. Liczę na odzew programistów po publikacji Kopry w Sklepie Microsoft. Zależy mi by inwestowanie przez Koprę było łatwą formą pomnażania oszczędności, bez płacenia wysokich prowizji pośrednikom, co pomoże poprawić domowy budżet wielu osób.





# Spis rysunków

2.1. Zestaw konfiguracji emulatorów dostępny w Windows Phone 8.1 SDK . . . . .	18
2.2. Zestaw konfiguracji emulatorów dostępny w Windows Mobile 10 SDK . . . . .	19
3.1. Struktura folderów w projekcie aplikacji Kopra . . . . .	23
3.2. Lista plików widoków w projekcie . . . . .	23
3.3. Lista plików widok-modeli w projekcie . . . . .	23
4.1. Logo na szerokim kafelku . . . . .	28
4.2. Logo na kwadratowym kafelku . . . . .	28
4.3. Logo na szerokim kafelku . . . . .	29



# Bibliografia

- [1] CHACON S., STRAUB B.: *Pro Git*, Apress, 2014, Wydanie II, ISBN 978-14-842-0077-3.
- [2] DINO E.: *Programming Microsoft ASP.NET 4*, Microsoft Press, 2011, ISBN 978-0-7356-4338-3.
- [3] JAJUGA K., JAJUGA T.: *Inwestycje. Instrumenty finansowe, ryzyko finansowe, inżynieria finansowa*, Warszawa, Wydawnictwo Naukowe PWN SA, 2015, Wydanie III zm., ISBN 978-83-01-14957-4.
- [4] DRABIK L., SOBOL E.: *Słownik języka polskiego PWN*, Wydawnictwo Naukowe PWN, 2007, ISBN 978-8-3011-7377-7.
- [5] Billionaire who broke the Bank of England,  
<http://www.telegraph.co.uk/finance/2773265/Billionaire-who-broke-the-Bank-of-England.html>,  
dostęp: 27.12.2016.
- [6] Design Universal Windows Platform (UWP) app,  
<https://dev.windows.com/en-us/design>,  
dostęp: 27.12.2016.
- [7] DoMore Archives,  
<http://blogs.microsoft.com/firehose/tag/DoMore/>,  
dostęp: 27.12.2016.
- [8] Financing Civilization,  
<http://vikingsom.yale.edu/will/finciv/chapter1.htm>,  
dostęp: 27.12.2016.
- [9] Fractional Reserve Banking - An Economist's Perspective (Transcript),  
<https://www.frbatlanta.org/education/classroom-economist/fractional-reserve-banking/economist>,  
dostęp: 27.12.2016.
- [10] Taking a Peek at Peer-to-Peer Lending,  
<http://business.time.com/2012/11/15/taking-a-peek-at-peer-to-peer-lending/>,  
dostęp: 27.12.2016.
- [11] Koniec lokat antybelkowych. Co w zamian,  
<http://www.money.pl/pieniadze/wiadomosci/arttykul/koniec;lokat;antybelkowych;co;w;zamian,>  
dostęp: 27.12.2016.
- [12] Wyrok NSA, sygn. II FSK 1472/10,  
<http://orzeczenia.nsa.gov.pl/doc/1F82F6552C>,  
dostęp: 27.12.2016.

- [13] Wyrok NSA, sygn. II FSK 142/10,  
<http://orzeczenia.nsa.gov.pl/doc/862AF4722F>,  
dostęp: 27.12.2016.
- [14] Oświadczenie Zespołu Kokos.pl w sprawie wyroku NSA o sygn. II FSK 1472/10,  
[https://kokos.pl/aktualnosci/czytaj?id=2012.03\\_06](https://kokos.pl/aktualnosci/czytaj?id=2012.03_06),  
dostęp: 27.12.2016.
- [15] Zostań Inwestorem – Kokos.pl,  
<https://kokos.pl/info/chce-inwestowac>,  
dostęp: 28.10.2016.
- [16] Podstawowe stopy procentowe NBP,  
<http://www.nbp.pl/home.aspx?f=/dzienne/stopy.htm>,  
dostęp: 27.12.2016.
- [17] Pożyczki społecznościowe finansowo.pl,  
<https://www.finansowo.pl>,  
dostęp: 27.12.2016.
- [18] Pożyczki społecznościowe zakra.pl,  
<https://zakra.pl>,  
dostęp: 27.12.2016.
- [19] Pożyczki społecznościowe kokos.pl,  
<https://kokos.pl>,  
dostęp: 27.12.2016.
- [20] Pożyczki społecznościowe lendico.pl,  
<https://www.lendico.pl>,  
dostęp: 27.12.2016.
- [21] Pożyczki społecznościowe zakramini.pl,  
<https://zakramini.pl>,  
dostęp: 27.12.2016.
- [22] Pożyczki społecznościowe applecredit.pl,  
<https://applecredit.pl>,  
dostęp: 27.12.2016.
- [23] Pożyczki społecznościowe capitalclub.pl,  
<https://capitalclub.pl>,  
dostęp: 27.12.2016.
- [24] Funkcje webAPI serwisu Zakra.pl,  
<https://zakra.pl/api>,  
dostęp: 27.12.2016.
- [25] Dokumentacja WebAPI serwisu Kokos.pl,  
<https://kokos.pl/webapiinfo/dokumentacja>,  
dostęp: 08.11.2015.
- [26] Microsoft Visual Studio Community 2015,  
<https://www.microsoft.com/en-us/download/details.aspx?id=48146>,  
dostęp: 27.12.2016.

- [27] Dokumentacja języka C# dla Visual Studio 2015,  
<https://msdn.microsoft.com/en-us/library/kx37x362.aspx>,  
dostęp: 27.12.2016.
- [28] Komponenty architektury zestawu narzędzi .NET,  
<https://docs.microsoft.com/en-us/dotnet/articles/standard/components>,  
dostęp: 27.12.2016.
- [29] Archiwum Windows SDK oraz emulatorów,  
<https://developer.microsoft.com/en-us/windows/downloads/sdk-archive>,  
dostęp: 27.12.2016.
- [30] Treść licencji MIT wykorzystywanej przez Newtonsoft.Json,  
<https://github.com/JamesNK/Newtonsoft.Json/blob/master/LICENSE.md>,  
dostęp: 27.12.2016.
- [31] Strona domowa projektu JsonPath,  
<http://goessner.net/articles/JsonPath/>,  
dostęp: 27.12.2016.
- [32] Plany abonamentowe serwisu GitHub.com,  
<https://github.com/pricing>,  
dostęp: 27.12.2016.
- [33] LINQPad as a Code Scratchpad,  
<http://www.linqpad.net/CodeSnippetIDE.aspx>,  
dostęp: 27.12.2016.
- [34] Opis LINQPad na Wikipedii,  
<https://en.wikipedia.org/wiki/LINQPad>,  
dostęp: 27.12.2016.
- [35] Repozytorium projektu ILSpy,  
<https://github.com/icsharpcode/ILSpy>,  
dostęp: 27.12.2016.
- [36] Oryginalna notka techniczna nt. wzorca MVC stworzona dla XEROX PARC w latach 1978-79,  
<http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html>,  
dostęp: 27.12.2016.