Studia i Materiały Informatyki Stosowanej, Tom 17, Nr 2, 2025
str. 33-37

SYSTEM RECENZOWANIA I KATALOGOWANIA FILMÓW NA PLATFORMACH MOBILNYCH

Karolina Staryszak, Krzysztof Tyburek

Uniwersytet Kazimierza Wielkiego
Wydział Informatyki
ul. Kopernika 1, 85-074 Bydgoszcz
e-mail: karolina.staryszak@student.ukw.edu.pl

Streszczenie:
Przedmiotem pracy jest wykonanie projektu oraz implementacja aplikacji mobilnej służącej do recenzowania filmów oraz umożliwiającej umieszczanie ich na listach przypisanych do konta użytkownika. W pracy umówiono wykorzystane technologie, wytyczne projektowe oraz ważne punkty dotyczące implementacji aplikacji.

Słowa kluczowe: *aplikacja mobilna, recenzje filmów, Android, API webowe, MySQL.*

SYSTEM FOR MOVIEREVIEWING AND CATALOGING ON MOBILE PLATFORMS

Abstract:
The aim of thisstudy was to design and implement a mobile application for reviewing and rating movies as well as managingmovielists. The articlediscusses the development process, technologiesused, and the three-tierarchitecture of the application. Particularattention was paid to challengesrelated to client-servercommunication and user data security.

Keywords:*mobile application, moviereviewing, Android, Web API, MySQL.*

1. Wstęp

W ostatnich latach dynamiczny rozwój usług wideo na żądanie oraz aplikacji mobilnych stworzył zapotrzebowanie na narzędzia ułatwiające użytkownikom organizację i ocenę treści filmowych. Dotychczasowe rozwiązania, mimo swojej popularności, często nie łączą funkcjonalności katalogowania filmów z możliwością ich recenzowania w jednym systemie, co stwarza przestrzeń do ulepszeń. Projekt aplikacji mobilnej na system Android ma na celu wypełnienie tej luki, oferując użytkownikom prosty w obsłudze interfejs umożliwiający przeglądanie filmów, tworzenie list, wystawianie ocen oraz dzielenie się opiniami.
Artykuł przedstawia proces projektowania i implementacji aplikacji, uwzględniając wybór technologii takich jak Android Studio, API webowe i MySQL. Omawia również potencjalne korzyści z jej wykorzystania w codziennym użytkowaniu oraz możliwości przyszłego rozwoju. Tematyka skupia się na zastosowaniu technologii mobilnych w zarządzaniu treściami multimedialnymi i tworzeniu narzędzi sprzyjających społecznościowym interakcjom.

2. Analiza wymagań

Proces analizy wymagań rozpoczął się od identyfikacji problemów użytkowników związanych z katalogowaniem i oceną filmów. Zdefiniowano wymagania funkcjonalne, obejmujące kluczowe funkcje aplikacji, takie jak:

- Rejestracja i logowanie użytkownika,
- Tworzenie oraz zarządzanie osobistymi listami filmów,
- Wystawianie ocen i pisanie recenzji,
- Przeglądanie list i recenzji innych użytkowników,
- Wyszukiwanie filmów i użytkowników.

Zdefiniowano również wymagania niefunkcjonalne, takie jak zastosowanie bezpiecznego szyfrowania haseł

(bcrypt), łatwość obsługi oraz responsywność interfejsu użytkownika. Wymagania te stanowiły podstawę do wyboru stosu technologicznego oraz wytycznych projektowych.

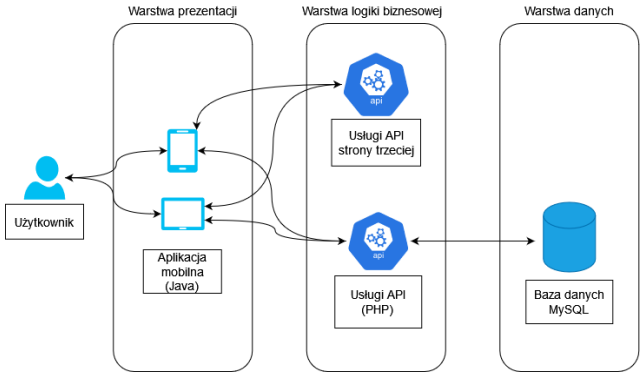
3. Modelowanie systemu

Do modelowania [8] systemu wykorzystano diagramy UML, które posłużyły do wizualizacji relacji pomiędzy komponentami oraz interakcji użytkownika z systemem.

Rys. 1. Diagram przypadków użycia

Diagram przypadków użycia zamieszczony na powyższym rysunku ilustruje kluczowe funkcjonalności aplikacji oraz interakcje użytkownika z systemem.Przedstawia scenariusze, takie jak dodawanie filmów na listę, wystawianie ocen oraz przeglądanie recenzji.
Zastosowano również diagramy architektury systemu, które ilustrowały trójwarstwową strukturę aplikacji [3]:

1. **Warstwa prezentacji** – aplikacja kliencka uruchamiana na urządzeniu mobilnym, umożliwiająca użytkownikowi interakcję z systemem.
2. **Warstwa logiki biznesowej** – serwer obsługujący operacje CRUD na bazie danych oraz realizujący logikę aplikacji za pośrednictwem API.
3. **Warstwa danych** – baza danych MySQL, przechowująca informacje o użytkownikach, filmach, listach i recenzjach.



Rys. 2. Architektura systemu.

Modelowanie systemu pozwoliło na precyzyjne określenie przepływu danych i komunikacji pomiędzy poszczególnymi komponentami aplikacji.

4. Projektowanie systemu

Na etapie projektowania [9] wybrano stos technologiczny i opracowano szczegółowe plany implementacyjne. Do stworzenia aplikacji wykorzystano:

- **Android Studio** jako środowisko programistyczne, z językiem Java do implementacji logiki aplikacji,
- **API webowe** do komunikacji pomiędzy aplikacją a bazą danych, z użyciem protokołu HTTP i formatu JSON,
- **Baza danych MySQL**, zarządzana za pomocą narzędzia MySQL Workbench, gdzie zaprojektowano strukturę tabel i relacji między nimi,
- **Biblioteka OkHttp** do realizacji asynchronicznych zapytań sieciowych,
- **Bcrypt** do bezpiecznego szyfrowania haseł użytkowników.

Podczas projektowania uwzględniono wzorce projektowe, takie jak Builder i Adapter, które usprawniły implementację złożonych komponentów i poprawiły czytelność kodu. Projekt interfejsu użytkownika został przygotowany w oparciu o układy XML, zapewniając intuicyjność i responsywność aplikacji.
Cały proces projektowy został podzielony na etapy: stworzenie prototypów interfejsu, implementację

podstawowych funkcji oraz integrację wszystkich elementów w finalną wersję aplikacji.

5. Baza danych

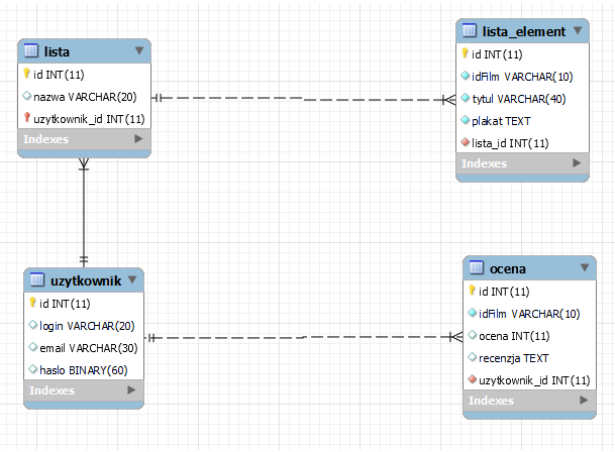
W projekcie zaimplementowano relacyjną bazę danych przy użyciu systemu MySQL, który zapewnia wydajną i spójną strukturę przechowywania danych. Projekt bazy obejmuje następujące elementy:

Zdefiniowano tabele dla użytkowników, ocen, list i elementów umieszczanych na listach. Tabela użytkowników przechowuje dane takie jak login, adres e-mail oraz hasło, które zostało zabezpieczone za pomocą algorytmu bcrypt, aby zwiększyć bezpieczeństwo danych.Filmy identyfikowane są przez unikalny identyfikator IMDB, a ich dodatkowe atrybuty to tytuł i plakat w postaci linku URL.Każda lista filmów jest powiązana z użytkownikiem oraz zawiera nazwę listy i przypisane do niej filmy.

Struktura bazy została zaprojektowana zgodnie z zasadami modelu relacyjnego:

- Każdy użytkownik może tworzyć wiele list filmów.
- Filmy mogą być powiązane z wieloma listami.
- Oceny i recenzje są przypisane zarówno do użytkownika, jak i do filmu, co pozwala na ich ścisłe powiązanie z konkretnym autorem oraz materiałem.

Diagram związków encji (EER) wykonano w MySQL Workbench, a kod SQL generowany automatycznie pozwala na łatwą implementację struktury. Zastosowano klucze główne, obce i unikalne, aby zapobiegać duplikacji oraz zapewnić spójność operacji w bazie.Baza danych została zoptymalizowana do działania na serwerze hostingu 000webhost, co umożliwia zdalny dostęp z aplikacji i sprawne przetwarzanie zapytań.



Rys. 3. Diagram EER bazy danych.

6. Implementacja

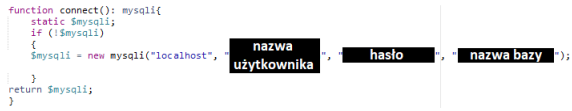
Jednym z podstawowych komponentów systemu Android są aktywności (activities), odpowiedzialne za interakcję z użytkownikiem [16]. Służą do tworzenia i otwierania okna aplikacji oraz uruchamiania innych komponentów.

Aplikacja składa się z aktywności odpowiedzialnych za:

- okno logowania,
- okno rejestracji,
- ekran główny,
- podgląd popularnych filmów,
- wyszukiwanie filmów,
- przeglądanie recenzji,
- przeglądanie list filmowych.

Dodatkowo wykorzystano klasę ArrayAdapter [12] do przekształcania obiektów klasy Movie na dane w kontrolce ListView, odpowiedzialnej za wyświetlanie listy filmowej.
Serwerowa część aplikacji opiera się na zestawie skryptów w języku PHP, które odpowiadają za logikę biznesową. Realizowane są tutaj operacje związane z komunikacją z bazą danych, takie jak pobieranie, zapisywanie czy aktualizacja danych.
Skrypty realizują operacje CRUD (Create, Read, Update,Delete). Wymiana danych między aplikacją a API odbywa się za pomocą protokołu HTTP, gdzie żądania przesyłane są w formacie JSON (GET lub POST).Przykładowo, zapytanie o filmy zapisane na liście użytkownika zwraca dane takie jak tytuł filmu, jego identyfikator oraz adres URL plakatu w formacie JSON.

Skrypty PHP łączą się z bazą danych MySQL, umieszczoną na tym samym serwerze przy użyciu dostępnego w PHP rozszerzenia MySQLi (MySQL Improved). Połączenie wygląda w sposób pokazany na rysunku:



Rys. 4. Funkcja PHP do łączenia z bazą

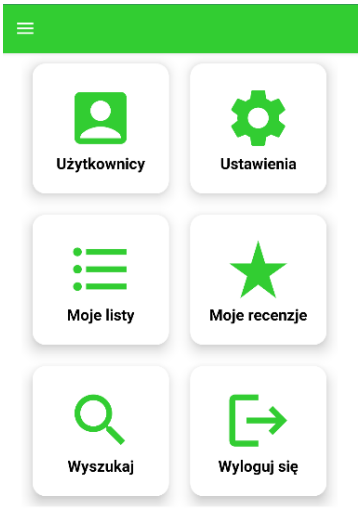
Do wysyłania zapytań HTTP wykorzystano bibliotekę OkHTTP. Wykorzystano jej możliwości do tworzenia asynchronicznych zapytań HTTP [18].Służy do tego funkcja enqueue() wywoływana na obiekcie typu Call. Żądanie jest wówczas wysłane w osobnym wątku i nie spowalnia pracy aplikacji, co jest szczególnie istotne, kiedy czas na otrzymanie odpowiedzi od serwera jest długi.

```
Call call = client.newCall(request);
call.enqueue(new Callback() {
    public void onResponse(Call call, Response response)
            throws IOException {
        if (response.isSuccessful()) {
            final String myResponse = response.body().string();

            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    try {

                                JSONObject jsonObject = new JSONObject(myResponse);
                                title= jsonObject.getString( name: "title");
                                tekst.setText(title);
                                summary=jsonObject.getString( name: "description");
                                textSummary.setText(summary);
                                imageURL=jsonObject.getString( name: "poster");
                                new FetchImage(imageURL,obrazek).start();


                    } catch (JSONException e) {
                        e.printStackTrace();
                    }
                }
            });

        }}

    public void onFailure(Call call, IOException e) {

    }
});
```

Rys. 5. Fragment funkcji asynchronicznej GET

5. Wyniki prac

W wyniku przeprowadzonych prac zaprojektowano i zaimplementowano aplikację mobilną na platformę Android, która umożliwia użytkownikom katalogowanie, ocenianie i recenzowanie filmów. Proces realizacji obejmował analizę wymagań, modelowanie systemu oraz projektowanie jego architektury i komponentów, co pozwoliło na precyzyjne odwzorowanie kluczowych funkcjonalności aplikacji.
Finalny produkt zapewnia użytkownikom możliwość rejestracji i logowania, przeglądania katalogu filmów, tworzenia i zarządzania listami, wystawiania recenzji oraz przeglądania opinii innych użytkowników. Dzięki trójwarstwowej architekturze aplikacja gwarantuje bezpieczeństwo danych i ich łatwy dostęp z różnych urządzeń. Wyniki prac potwierdzają, że zaprojektowane rozwiązanie spełnia założenia funkcjonalne, a jego potencjalny rozwój może objąć dodatkowe elementy społecznościowe oraz usprawnienia technologiczne.



Rys. 6. Ekran główny aplikacji

6. Podsumowanie i wnioski

Przeprowadzony projekt pozwolił na stworzenie aplikacji mobilnej, która skutecznie łączy funkcjonalności katalogowania filmów z możliwością ich oceniania i recenzowania. Główne cele zostały osiągnięte – aplikacja oferuje intuicyjny interfejs użytkownika, integrację z bazą danych, bezpieczeństwo przechowywania danych oraz możliwość korzystania z publicznych API do pozyskiwania informacji o filmach. Mocną stroną projektu jest zastosowanie sprawdzonych technologii, takich jak Android Studio, MySQL oraz biblioteka OkHttp, co zapewnia niezawodność i możliwość łatwego rozwijania aplikacji w przyszłości.
Do zalet należy także przejrzysta architektura trójwarstwowa, która gwarantuje ochronę danych i modularność systemu. Szczególną uwagę poświęcono aspektowi bezpieczeństwa – implementacja szyfrowania haseł z użyciem bcrypt chroni użytkowników przed potencjalnymi zagrożeniami. Warto podkreślić, że aplikacja spełnia potrzeby zarówno indywidualnych

użytkowników, jak i grup społecznościowych, umożliwiając interakcje poprzez wymianę opinii i podgląd list innych użytkowników.

Niektóre ograniczenia projektu wynikają z zastosowania darmowych publicznych API, które oferują ograniczoną ilość wyników i mogą nie być wystarczające przy większej liczbie użytkowników. Ponadto użycie darmowego hostingu serwera, choć zaspokaja obecne potrzeby, może nie być optymalnym rozwiązaniem w przypadku znacznego wzrostu ruchu na platformie.

Znaczenie projektu jest istotne zarówno dla użytkowników indywidualnych, jak i szerszego pola badań nad aplikacjami społecznościowymi. Aplikacja nie tylko odpowiada na współczesne potrzeby organizacji treści, ale także pokazuje, jak efektywna może być integracja funkcji społecznościowych z personalizowanymi rozwiązaniami.

Możliwości rozwoju obejmują wprowadzenie dodatkowych funkcji, takich jak moderacja recenzji, reakcje i komentarze do list oraz recenzji, czy rozbudowanie aspektów społecznościowych, np. o listę znajomych. W zakresie technologii warto rozważyć zmianę dostawcy API na bardziej zaawansowanego oraz migrację na lepszy hosting serwera, co pozwoliłoby na obsługę większej liczby użytkowników i lepszą wydajność systemu.

Projekt dowodzi, że odpowiednio zaplanowana i wykonana aplikacja może skutecznie zaspokajać potrzeby użytkowników, a jego potencjał rozwoju otwiera drogę do nowych możliwości w obszarze aplikacji multimedialnych.

Literatura

1. Fiona D.M.C.S., "UML – Zarys", dostępne online: http://fiona.dmcs.p.lodz.pl/epjc/Wyklad/UML%20-%20zarys.pdf
2. Wolski P., "Diagramy UML – Przypadki użycia", dostępne online: https://wolski.pro/diagramy-uml/diagram-przypadkw-uzycia/
3. IBM Documentation, "Three-TierArchitectures", dostępne online: https://www.ibm.com/docs/pl/was/9.0.5?topic=overview-three-tier-architectures
4. Blog Desdelinux, "Cechy i charakterystyki Android Studio", dostępne online: https://blog.desdelinux.net/pl/cechy-i-cechy-Android-Studio/
5. Block G., "Nowoczesne API. Ewoluujące aplikacje sieciowe w technologii ASP.NET", Helion.
6. Jacobson D., Woods D., Brail G., "Interfejs API: Strategia programisty", Gliwice: Helion, 2015.
7. Gilmore W.J., "PHP i MySQL. Od podstaw. Wydanie IV", Helion.
8. Brasil C.E.L., "Modelowanie systemu", dostępne online: https://brasil.cel.agh.edu.pl/~10sdczerner/page/modelowanie_systemu.html
9. Brasil C.E.L., "Projektowanie systemu", dostępne online: https://brasil.cel.agh.edu.pl/~10sdczerner/page/projektowanie_systemu.html
10. Refactoring Guru, "Klasyfikacja wzorców projektowych", dostępne online: https://refactoring.guru/pl/design-patterns/classification
11. CodePathGuides, "Using anArrayAdapter with ListView", dostępne online: https://guides.codepath.com/android/Using-an-ArrayAdapter-with-ListView
12. Android Developers, "ArrayAdapter", dostępne online: https://developer.android.com/reference/android/widget/ArrayAdapter
13. Refactoring Guru, "Builder – Wzorzec projektowy", dostępne online: https://refactoring.guru/pl/design-patterns/builder
14. Tutorials Point, "Android Alert Dialogs", dostępne online: https://www.tutorialspoint.com/android/android_alert_dialoges.htm
15. SquareOkHttp, "Request.BuilderDocumentation", dostępne online: https://square.github.io/okhttp/3.x/okhttp/okhttp3/Request.Builder.html
16. Strefa Kodera, "Aktywności w Androidzie", dostępne online: https://strefakodera.pl/programowanie/android-java/aktywnosci-w-androidzie
17. Android Developers, "DeclaringLayouts", dostępne online: https://developer.android.com/develop/ui/views/layout/declaring-layout
18. SquareOkHttp, "Recipes", dostępne online: https://square.github.io/okhttp/recipes/
19. Burdelak D., "Bcrypt – Bezpieczne przechowywanie hasła w bazie danych", dostępne online: https://davidburdelak.pl/blog/bcrypt-bezpieczne-przechowywanie-hasla-w-bazie-danych
20. Android Developers, "ViewBinding", dostępne online: https://developer.android.com/topic/libraries/view-binding