

Wstęp

Twitter to jedna z głównych platform społecznościowych, która zmieniła sposób komunikowania się i dzielenia informacjami w czasach cyfrowych. Jako student, który jest zainteresowany aplikacjami mobilnymi i programowaniem, naturalnym wyborem było dla mnie podjęcie projektu polegającego na odtworzeniu funkcjonalności Twittera.

Tworzenie klona aplikacji Twittera od podstaw dało mi okazję nie tylko do pogłębienia wiedzy z zakresu tworzenia aplikacji na Androida, ale także pozwoliło mi zapoznać się z szeregiem nowych technologii potrzebnych do budowy pełni funkcjonalnej aplikacji społecznościowej.

Ponieważ było to moje pierwsze podejście do dostępu do zewnętrznej bazy danych przy użyciu interfejsów API, wybrałem technologie PHP i MySQL jako moją podstawę aby zacząć. Było to uzasadnione wyborem biorąc pod uwagę moją znajomość tych technologii na ówczesnym etapie.

Celem tego projektu inżynierskiego jest stworzenie w pełni funkcjonalnej aplikacji internetowej w stylu Twittera, do użytku prywatnego dla rodziny i znajomych, w której miałbym pełną kontrolę jako administrator. Aplikacja będzie posiadać podstawowe funkcje mediów społecznościowych takie jak tworzenie postów, polubienia, komentarze i obserwowanie innych użytkowników. Zostanie ona zbudowana jako natywna aplikacja Androida przy użyciu Javy jako głównego języka programowania i Android Studio jako podstawowego środowiska IDE.

Backend będzie oparty o PHP do tworzenia punktów końcowych API, które będą komunikować się z bazą danych MySQL hostowaną w chmurze. Pomimo rozwoju nowszych rozwiązań backendowych, PHP i MySQL są nadal popularną i sprawdzoną technologią dla wielu aplikacji internetowych.

W następnych rozdziałach tej pracy inżynierskiej przedstawię teoretyczne podstawy programowania aplikacji Androida oraz koncepcje mediów społecznościowych istotne dla tego projektu. Następnie omówię analizę wymagań systemowych, architekturę aplikacji i projekt bazy danych. W dalszej części opiszę szczegóły implementacji kluczowych funkcji, napotkane wyzwania i rozwiązania. Wreszcie, pokażę wyniki testów aplikacji oraz wnioski i potencjalne kierunki dalszego rozwoju.

Budując tę aplikację, moim celem było nie tylko stworzenie praktycznego narzędzia do użytku osobistego, ale przede wszystkim pogłębienie wiedzy i umiejętności w zakresie rozwoju aplikacji mobilnych. Mam nadzieję, że projekt ten będzie odzwierciedlał mój rozwój jako

inżyniera oprogramowania i stanowił solidny fundament do dalszej nauki i doskonalenia w tej dziedzinie.

Podstawy teoretyczne

Rozwój aplikacji na Androida

Android to otwartoźródłowy system operacyjny dla urządzeń mobilnych opracowany przez Google, który zyskał ogromną popularność na całym świecie. Programowanie aplikacji na Androida odbywa się głównie w języku Java, choć nowsze narzędzia takie jak Kotlin zyskują na popularności. Głównym IDE (zintegrowanym środowiskiem programistycznym) dla programistów Androida jest Android Studio, oferujące bogaty zestaw narzędzi do budowania, testowania i debugowania aplikacji.

Aplikacje Androida są zbudowane z różnych komponentów, takich jak aktywności (activities), usługi (services), dostawcy treści (content providers) i odbiorcy rozgłoszeń (broadcast receivers). Aktywności reprezentują pojedynczy ekran z interfejsem użytkownika, podczas gdy usługi działają w tle, wykonując długotrwałe operacje. Dostawcy treści zarządzają współdzielonym zestawem danych aplikacji, a odbiorcy rozgłoszeń pozwalają aplikacji na reakcję na ogłoszenia rozgłoszeń systemowych[1][2].

Interfejs użytkownika w aplikacjach Androida jest zbudowany przy użyciu hierarchii obiektów View i ViewGroup. Widoki (View) są podstawowymi elementami składającymi się na interfejs użytkownika, takimi jak przyciski, pola tekstowe czy obrazy. Grupy widoków (ViewGroup) są niewidzialnymi kontenerami, które definiują strukturę układu dla widoków i innych grup widoków[3].

Android dostarcza wielu wbudowanych widoków i kontrolerek, ale programiści mogą także tworzyć własne niestandardowe widoki według potrzeb. Układ interfejsu użytkownika jest zazwyczaj definiowany w plikach XML, podczas gdy interakcje są obsługiwane w kodzie Javy.

Jednym z kluczowych aspektów programowania Androida jest cykl życia komponentów, szczególnie aktywności. Aktywności mają określone stany, takie jak Wznowiona (Resumed), Wstrzymana (Paused), Zatrzymana (Stopped) i inne. Prawidłowe zarządzanie cyklem życia aktywności jest niezbędne do tworzenia responsywnych i wydajnych aplikacji[2][4].

Android oferuje również różne mechanizmy do przechowywania danych, takie jak SharedPreferences, pliki, bazy danych SQLite i połączenia sieciowe. W tym projekcie skupimy się na komunikacji z zewnętrzną bazą danych MySQL za pomocą interfejsów API PHP.

Koncepcje mediów społecznościowych

Media społecznościowe zrewolucjonizowały sposób, w jaki ludzie komunikują się, dzielą informacjami i wchodzą w interakcje w Internecie. Platformy takie jak Twitter, Facebook czy Instagram zyskały ogromną popularność na całym świecie, przyciągając miliony, a czasem nawet miliardy użytkowników.

Jednym z kluczowych aspektów mediów społecznościowych jest tworzenie i udostępnianie treści. Użytkownicy mogą publikować teksty, obrazy, filmy czy linki, wyrażając swoje myśli, doświadczenia i opinie. W przypadku Twittera, te treści przyjmują formę krótkich wiadomości zwanych tweetami, które są ograniczone do 280 znaków[5].

Innym ważnym elementem jest interakcja między użytkownikami. Media społecznościowe umożliwiają użytkownikom polubienie, komentowanie i udostępnianie treści innych osób. Tworzy to dynamiczne dyskusje i pozwala na rozprzestrzenianie się informacji w sieciach społecznych[6].

Koncepcja obserwowania (following) jest kolejnym fundamentalnym aspektem wielu platform społecznościowych. Użytkownicy mogą subskrybować lub "obserwować" innych użytkowników, których treści chcą regularnie widzieć. Na Twitterze to prowadzi do wyświetlania tweetów obserwowanych użytkowników na stronie głównej, zwanej timeline[7].

Media społecznościowe wprowadziły także koncepcję tagowania i hashtagów. Użytkownicy mogą oznaczać innych w swoich postach za pomocą symbolu @ po którym następuje nazwa użytkownika. Hashtagi, z kolei, to słowa kluczowe poprzedzone symbolem #, które pozwalają na kategoryzację treści i ułatwiają wyszukiwanie postów na dany temat[5][8].

W kontekście rozwoju aplikacji społecznościowych, ważne jest, aby uwzględnić te kluczowe koncepcje i funkcje. W tym projekcie skupię się na podstawowych elementach, takich jak publikowanie postów, polubienia, komentarze i obserwowanie innych użytkowników.

Wykorzystane technologie

Java

Java to obiektowy język programowania ogólnego przeznaczenia, który jest szeroko stosowany w rozwoju aplikacji Androida. Została wybrana jako główny język dla tego projektu ze względu na moją uprzednią znajomość Javy z zajęć Programowania Obiektowego na studiach oraz jej rozległe wsparcie w ekosystemie Androida.

Jedną z kluczowych zalet Javy jest jej przenośność. Kod Javy jest kompilowany do kodu bajtowego, który może być uruchamiany na dowolnej maszynie wirtualnej Javy (JVM), niezależnie od podstawowej architektury sprzętowej. Ta zasada "napisz raz, uruchamiaj wszędzie" jest szczególnie przydatna w przypadku programowania aplikacji mobilnych, która muszą działać na różnych urządzeniach[9][10].

Java jest w pełni zorientowana obiektowo, co oznacza, że wszystko (z wyjątkiem typów prymitywnych) jest obiektem. Zapewnia to czystą i modułową strukturę kodu. Java obsługuje również wiele zaawansowanych koncepcji, takich jak dziedziczenie, polimorfizm, abstrakcja i hermetyzacja[11].

Java ma rozległe biblioteki standardowe, które dostarczają gotowych narzędzi do wielu typowych zadań programistycznych, takich jak I/O, obsługa sieci, struktury danych czy współbieżność. Do budowy interaktywnych aplikacji z graficznym interfejsem użytkownika Java oferuje biblioteki takie jak AWT, Swing i JavaFX[12] - choć w przypadku Androida używany jest dedykowany framework do tworzenia interfejsów.

Pomimo pojawienia się nowszych języków takich jak Kotlin, który jest mocno promowany przez Google dla programowania Androida, Java pozostaje niezwykle popularnym i szeroko wspieranym wyborem. Większość istniejących aplikacji na Androida jest napisanych w Javie, a jej dojrzałe środowisko i dostępność zasobów sprawiają, że jest to solidny wybór szczególnie dla początkujących programistów.

PHP

PHP (rekurencyjny akronim od PHP: Hypertext Preprocessor) jest szeroko stosowanym językiem skryptowym open-source ogólnego przeznaczenia. Jest szczególnie dostosowany do programowania webowego i może być osadzony w HTML[13].

Jedną z kluczowych zalet PHP jest jego prostota dla nowych programistów i jednocześnie oferowanie wielu zaawansowanych funkcji dla profesjonalnych programistów. Z PHP można robić wszystko, od prostych skryptów po złożone aplikacje webowe[13][14].

PHP jest językiem dynamicznie typowanym i interpretowanym, co oznacza, że typowanie zmiennych jest decydowane w czasie wykonania, a kod jest wykonywany bezpośrednio bez wcześniejszej kompilacji. Ma to swoje zalety, takie jak szybkie prototypowanie, ale może też prowadzić do pewnych problemów wynikających z braku statycznej analizy kodu[15].

PHP ma rozbudowane wsparcie do interakcji z bazami danych, szczególnie MySQL. Posiada wbudowane funkcje do łączenia się z bazami danych, wykonywania zapytań i obsługi wyników. Umożliwia to łatwe tworzenie dynamicznych stron internetowych zasilanych danymi z bazy[16].

Inną ważną cechą jest wbudowane wsparcie dla protokołu HTTP i przetwarzanie danych z formularzy. PHP może łatwo pobierać dane przesłane przez użytkownika za pomocą metod GET i POST, a także przekazywać dane między stronami za pomocą ciasteczek i sesji[14][17].

PHP posiada również szerokie możliwości manipulowania tekstem, obsługi ekspresji regularnych, generowania obrazów, pracy z różnymi formatami plików. Są to przydatne funkcje przy budowaniu różnorodnych aplikacji webowych[13].

W kontekście tego projektu zdecydowałem się użyć PHP do stworzenia interfejsów API po stronie serwera. Endpointy API będą odbierać żądania HTTP z aplikacji Androida, przetwarzać je, wchodzić w interakcję z bazą danych MySQL i zwracać odpowiedzi w formacie JSON. Ta architektura pozwala na oddzielenie logiki frontendu i backendu aplikacji.

MySQL

MySQL to najpopularniejszy system zarządzania relacyjnymi bazami danych typu open source. Jest szeroko stosowany w aplikacjach webowych w połączeniu z PHP[18].

Jedną z kluczowych zalet MySQL jest jego wydajność i skalowalność. Potrafi obsłużyć duże ilości danych i dużą liczbę równoczesnych użytkowników. Jest to kluczowe przy budowaniu aplikacji takich jak media społecznościowe, które mogą mieć do czynienia z dużym ruchem[19].

MySQL oferuje standardowe funkcje SQL do definiowania struktury danych, wykonywania zapytań, wstawiania, aktualizowania i usuwania rekordów. Zapewnia także zaawansowane funkcje takie jak transakcje, procedury składowane, triggerzy czy widoki[20].

Jedną z mocnych stron MySQL jest jego wsparcie dla różnych silników przechowywania tabel, z których najpopularniejsze to InnoDB i MyISAM. InnoDB jest domyślnym silnikiem i oferuje obsługę transakcji ACID, obsługę kluczy obcych i blokowanie na poziomie wierszy. MyISAM jest starszym silnikiem, który nie obsługuje transakcji, ale w niektórych scenariuszach może oferować większą wydajność[18][21].

MySQL oferuje także wbudowane funkcje bezpieczeństwa, takie jak uwierzytelnianie użytkowników, uprawnienia dostępu i szyfrowanie połączeń. Jest to niezwykle istotne przy budowaniu aplikacji internetowych, aby chronić dane użytkowników[20].

W kontekście tego projektu MySQL będzie używany do przechowywania wszystkich danych aplikacji, takich jak informacje o użytkownikach, tweety, polubienia i komentarze. Baza danych będzie hostowana online, a komunikacja z aplikacją Androida będzie odbywać się poprzez API oparte na PHP.

Wybór hosting MySQL online zamiast lokalnej bazy danych jest uzasadniony potrzebami tego projektu. Ponieważ aplikacja ma być dostępna dla mojej rodziny i przyjaciół, baza danych musi być dostępna przez internet. Hostowanie jej na niezawodnym dostawcy zapewnia dostępność i skalowalność, niezbędne dla tego typu aplikacji.

W tej sekcji omówiłem teoretyczne podstawy programowania aplikacji na Androida oraz koncepcje mediów społecznościowych. Przedstawiłem także kluczowe technologie użyte w tym projekcie - Javę do programowania aplikacji klienckiej, PHP do budowy API i MySQL jako system zarządzania bazą danych. W kolejnym rozdziale przejdę do analizy i projektowania systemu.

Źródła:

[1] <https://developer.android.com/guide/components/fundamentals>: Fundamentals of Android app development.

[2] <https://www.udemy.com/course/android-app-development-course/>: Android App Development Course.

[3] <https://developer.android.com/guide/topics/ui/declaring-layout>: Layouts in Android.

- [4] <https://developer.android.com/guide/components/activities/activity-lifecycle>: The Activity Lifecycle in Android.
- [5] <https://blog.hootsuite.com/social-media-definitions/>: 23+ Social Media Definitions You Need to Know in 2023.
- [6] <https://www.interaction-design.org/literature/topics/social-media>: Social Media in Design - A 20 minute read.
- [7] <https://help.twitter.com/en/using-twitter/following-faqs>: FAQs about following on Twitter.
- [8] <https://help.twitter.com/en/using-twitter/how-to-use-hashtags>: How to use hashtags on Twitter.
- [9] <https://www.programiz.com/java-programming>: Learn Java Programming – Programiz.
- [10] https://www.w3schools.com/java/java_intro.asp: Java Introduction – W3Schools.
- [11] https://www.tutorialspoint.com/java/java_quick_guide.htm: Java Quick Guide – Tutorialspoint. [12] <https://docs.oracle.com/javase/8/javase-clienttechnologies.htm>: Java SE 8 Documentation - Client Technologies.
- [13] <https://www.php.net/manual/en/intro-what-is.php>: What is PHP? – PHP.net.
- [14] https://www.w3schools.com/php/php_intro.asp: PHP Introduction – W3Schools.
- [15] <https://www.geeksforgeeks.org/php-data-types/>: PHP Data Types – GeeksforGeeks.
- [16] <https://www.php.net/manual/en/book.mysql.php>: MySQL PHP API – PHP.net.
- [17] <https://www.php.net/manual/en/book.http.php>: HTTP Extension – PHP.net.
- [18] <https://www.mysql.com/why-mysql/>: Why MySQL? – MySQL.com.
- [19] <https://www.guru99.com/introduction-to-mysql.html>: MySQL Tutorial for Beginners – Guru99. [20] <https://dev.mysql.com/doc/refman/8.0/en/features.html>: MySQL 8.0 Reference Manual – MySQL. Features
- [21] <https://dev.mysql.com/doc/refman/8.0/en/storage-engines.html>: MySQL 8.0 Reference Manual - Storage Engines.

Rozdział 3: Analiza i projektowanie systemu

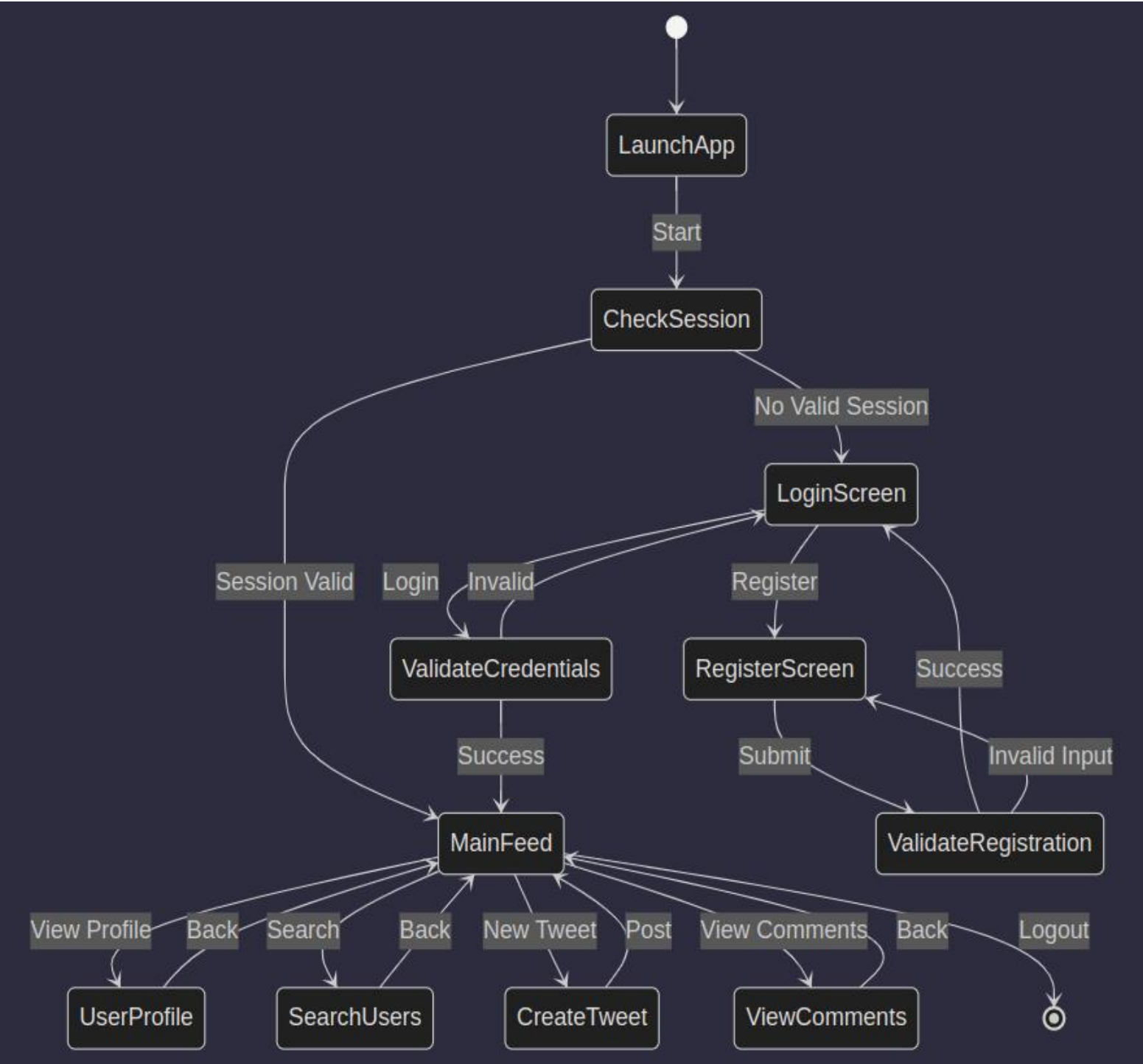
Wymagania systemowe

Pierwszym krokiem w procesie tworzenia aplikacji jest zdefiniowanie jej wymagań. W przypadku tego projektu klona Twittera na Androida, głównym celem jest zbudowanie aplikacji z podstawowymi funkcjami mediów społecznościowych dla prywatnego użytku rodziny i przyjaciół. Na podstawie tego założenia, zdefiniowałem następujące kluczowe wymagania:

1. Rejestracja i logowanie użytkownika: Użytkownicy powinni mieć możliwość utworzenia nowego konta i zalogowania się do aplikacji przy użyciu adresu e-mail i hasła.
2. Tworzenie i publikowanie tweetów: Zalogowani użytkownicy powinni mieć możliwość tworzenia nowych tweetów o ograniczonej długości i publikowania ich do globalnej osi czasu.
3. Polubienia i komentarze: Użytkownicy powinni mieć możliwość polubienia i skomentowania tweetów innych użytkowników. Liczba polubień i komentarzy powinna być widoczna pod każdym tweetem.
4. Podążanie za użytkownikami: Użytkownicy powinni mieć możliwość "podążania" za innymi użytkownikami, aby widzieć ich tweety na swojej stronie głównej.
5. Strona główna: Główny ekran powinien zawierać listę tweetów od obserwowanych użytkowników, posortowaną od najnowszych.
6. Profil użytkownika: Każdy użytkownik powinien mieć własny profil pokazujący jego tweety, liczbę obserwujących i obserwowanych.

Przypadki użycia i wymagania funkcjonalne

Jednym z kluczowych wymagań funkcjonalnych mojej aplikacji klona Twittera jest proces uwierzytelniania użytkownika. Obejmuje on rejestrację nowych użytkowników, logowanie istniejących użytkowników oraz zarządzanie sesjami.



Powyższy diagram przedstawia przepływ uwierzytelniania użytkownika w mojej aplikacji:

1. Użytkownik uruchamia aplikację i jest kierowany na ekran logowania (LoginScreen).
2. Jeśli użytkownik nie ma jeszcze konta, może przejść do ekranu rejestracji (RegisterScreen), gdzie podaje wymagane dane, takie jak nazwa użytkownika, adres e-mail i hasło. Po pomyślnej walidacji danych (ValidateRegistration), konto jest tworzone, a użytkownik jest przenoszony do ekranu głównego (MainFeed).
3. Jeśli użytkownik posiada konto, wprowadza swoje dane logowania na ekranie logowania. System waliduje podane poświadczenia (ValidateCredentials). Jeśli są one poprawne, tworzona jest nowa sesja (Session Valid), a użytkownik jest kierowany na ekran główny. Jeśli poświadczenia są nieprawidłowe (Invalid), użytkownik pozostaje na ekranie logowania.
4. Na każdym chronionym ekranie (np. MainFeed, UserProfile, SearchUsers itp.), system sprawdza, czy użytkownik ma ważną sesję (CheckSession). Jeśli sesja jest prawidłowa, użytkownik może korzystać z funkcjonalności aplikacji. Jeśli sesja jest nieprawidłowa (No Valid Session), użytkownik jest przekierowywany z powrotem na ekran logowania.

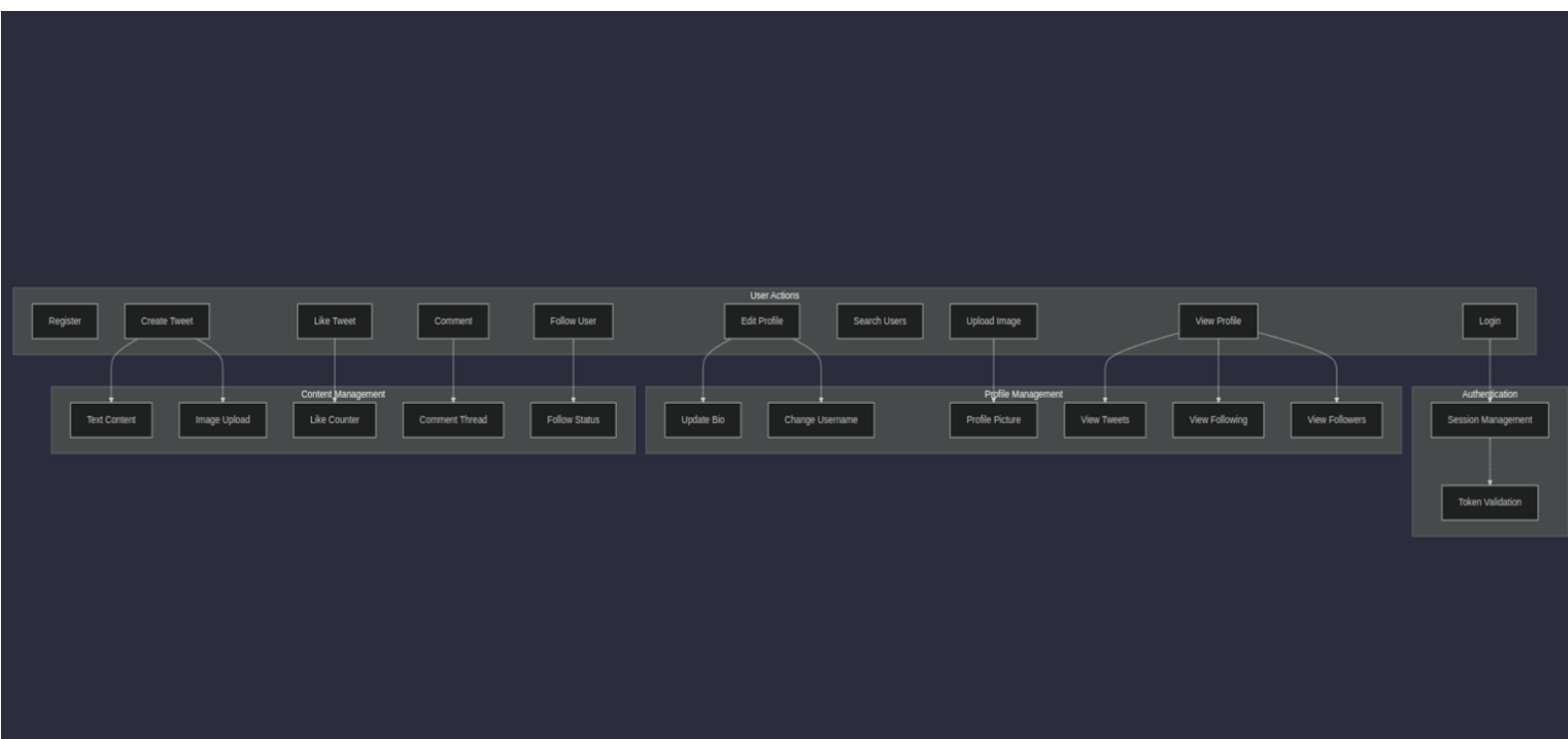
Ten przepływ uwierzytelniania zapewnia, że tylko uwierzytelnieni użytkownicy mogą uzyskać dostęp do kluczowych funkcji aplikacji, takich jak przeglądanie postów, tworzenie tweetów czy interakcja z innymi użytkownikami.

Implementacja tego przepływu wymagała starannej koordynacji między aplikacją frontendową (Androidem) a endpointami API backendu. Musiałem zapewnić, że dane użytkownika są bezpiecznie przesyłane i walidowane, a stany sesji są poprawnie zarządzane.

Chociaż początkowo znalezienie odpowiedniej równowagi między bezpieczeństwem a użytecznością było wyzwaniem, uważam, że końcowy rezultat spełnia założone wymagania funkcjonalne. Użytkownicy mogą płynnie rejestrować się i logować do aplikacji, podczas gdy system zapewnia niezbędne kontrole bezpieczeństwa.

Oczywiście, przepływ ten można dalej ulepszać. Na przykład, można zaimplementować funkcję "przypomnij hasło" dla użytkowników, którzy zapomną swoich danych logowania. Można również rozważyć dodanie opcji "zapamiętaj mnie" dla wygody użytkowników.

Mimo tych potencjalnych ulepszeń, jestem zadowolony z tego, jak obecny proces uwierzytelniania spełnia kluczowe wymagania funkcjonalne zdefiniowane dla mojej aplikacji klona Twittera. Daje on solidne podstawy, na których można budować bardziej zaawansowane funkcjonalności.



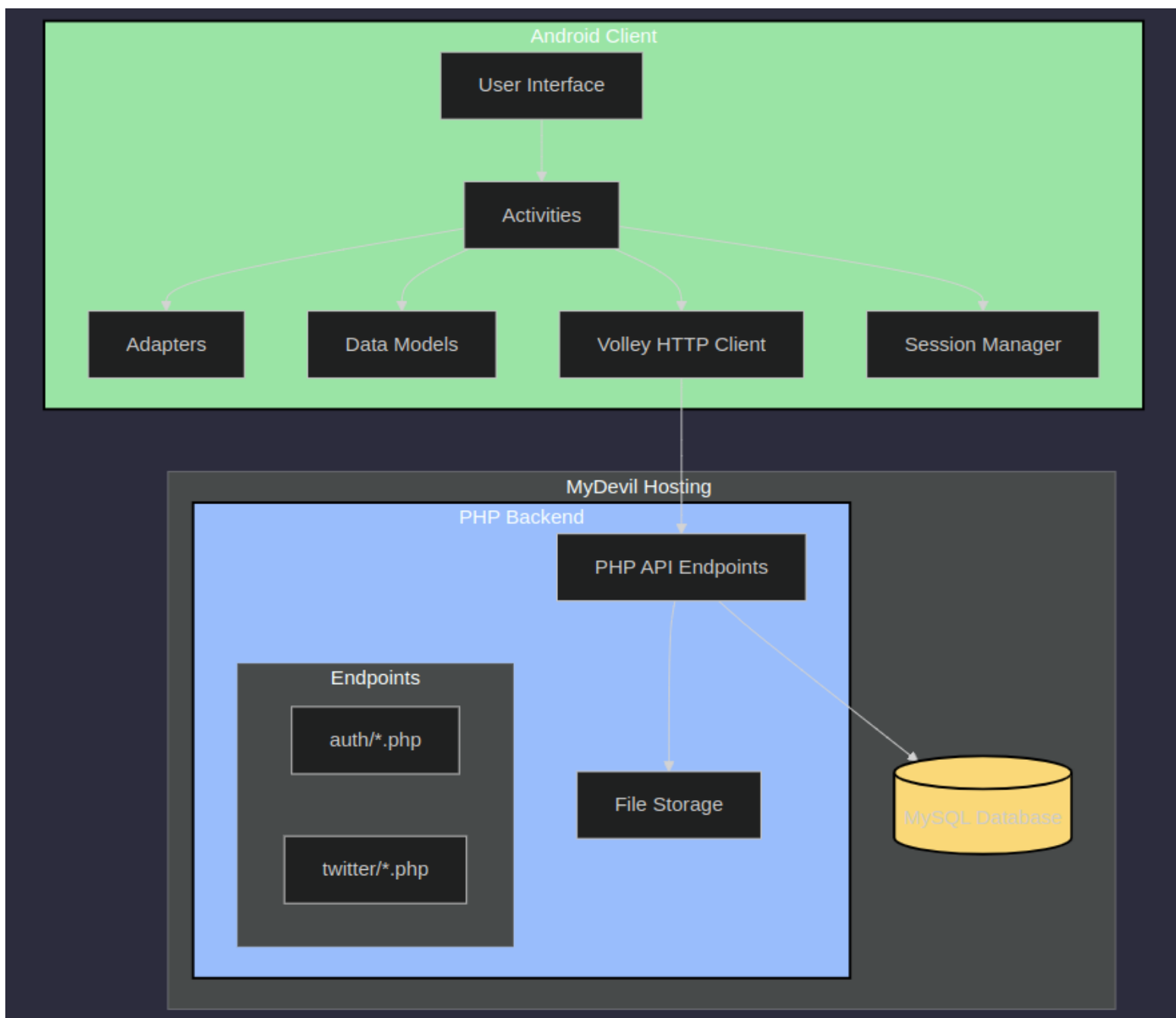
usecase_diagram_screenshot

Oprócz podstawowej funkcjonalności, ważnym aspektem było zaprojektowanie interfejsu użytkownika w sposób przyjazny dla użytkownika i dostosowany do ekranów urządzeń mobilnych różnych rozmiarów. Aplikacja powinna mieć czysty i intuicyjny układ, z łatwym dostępem do kluczowych funkcji.

Architektura systemu

Następnym krokiem po zdefiniowaniu wymagań jest zaprojektowanie architektury systemu. W przypadku tej aplikacji zdecydowałem się na architekturę klient-serwer, gdzie aplikacja Androida pełni rolę klienta, a serwer PHP z bazą danych MySQL działa jako backend.

Oto ogólny schemat architektury:



architecture_diagram_screenshot

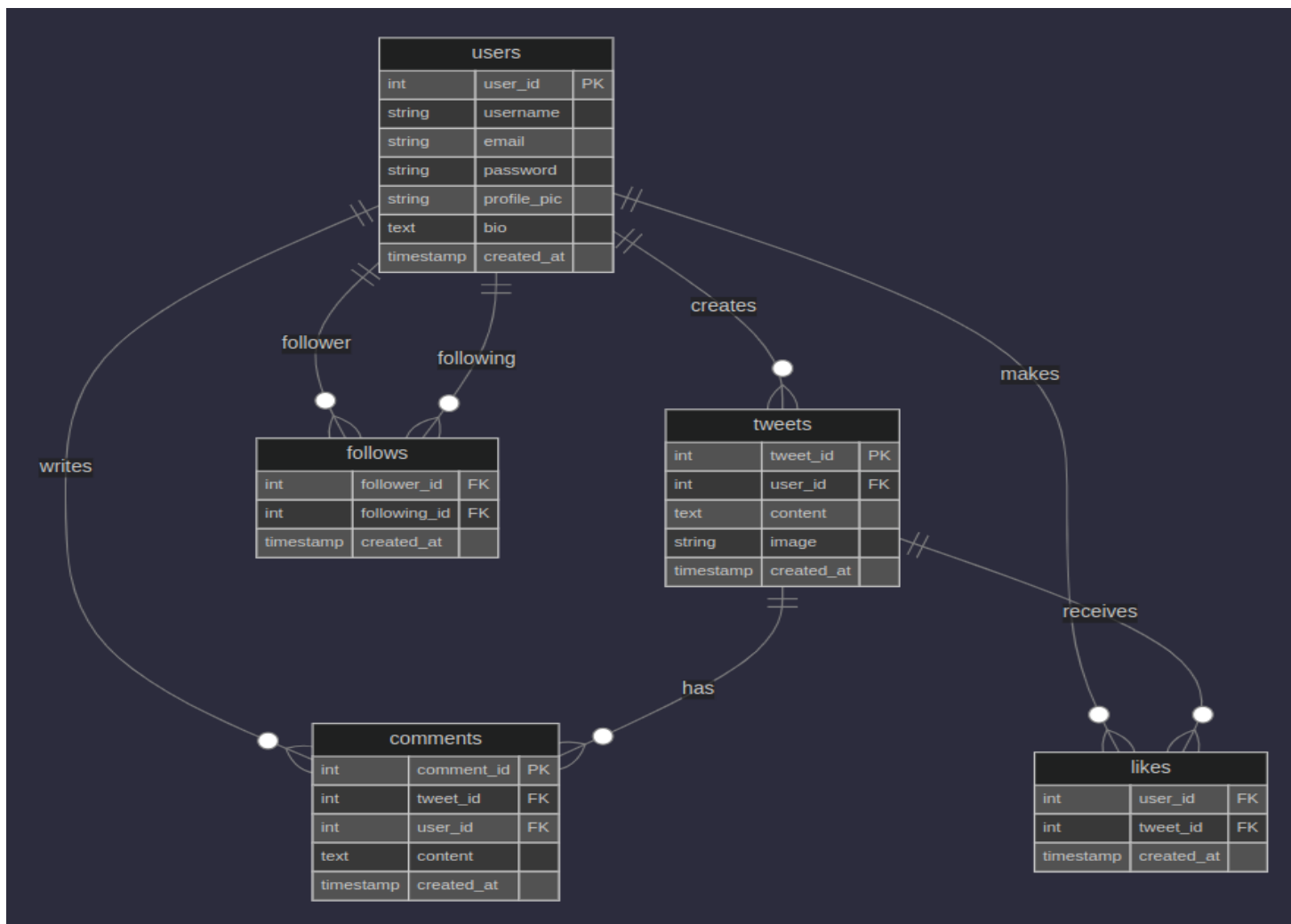
Aplikacja Androida będzie komunikować się z serwerem za pomocą żądań HTTP do endpointów API. Serwer będzie przetwarzać te żądania, wykonywać operacje na bazie danych i zwracać odpowiedzi w formacie JSON.

Taka architektura pozwala na jasne oddzielenie logiki po stronie klienta od logiki po stronie serwera. Aplikacja Androida jest odpowiedzialna za renderowanie interfejsu użytkownika,

interakcje i wysyłanie żądań, podczas gdy serwer obsługuje logikę biznesową, uwierzytelnianie, autoryzację i trwałość danych.

Projekt Bazy Danych

Baza danych jest kluczowym elementem aplikacji, ponieważ przechowuje wszystkie dane użytkowników i treści. Na podstawie wymagań zaprojektowałem następujący schemat relacyjnej bazy danych:



database_diagram_screenshot

Główne tabele w bazie danych to:

1. **Users**: przechowuje informacje o użytkownikach, takie jak nazwa użytkownika, adres e-mail, hasło i zdjęcie profilowe.

phpMyAdmin

Serveur courant : mysql33.mydevil.net

Récentes

Préférées

Nouvelle base de données

m1039_blog

Nouvelle table

comments

follows

likes

tweets

users

Server : mysql33.mydevil.net > Base de données : m1039_blog > Table : users

Parcourir

Structure

SQL

Rechercher

Insérer

Exporter

Importer

Opérations

Suivi

Déclencheurs

Profilage

Éditer en ligne

Éditer

Expliquer SQL

Créer le code source PHP

Actualiser

☐ Tout afficher

Nombre de lignes : 25

Filtrer les lignes: Chercher dans cette table

Trier par clé : Aucun(e)

Options supplémentaires

				user_id	username	email	password	profile_pic	bio	created_at
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	john_doe	john@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_1.jpg	Tech enthusiast	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	alice_smith	alice@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_2.jpg	Travel lover	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	dev_mike	mike@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_3.jpg	Software Developer	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	sarah_j	sarah@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_4.jpg	Photography passion	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	tech_guru	guru@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_5.jpg	Tech news and reviews	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	foodie_anna	anna@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_6.jpg	Food explorer	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	gamer_x	gamer@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_7.jpg	Gaming is life	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	fitness_pro	fitness@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_8.jpg	Fitness trainer	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	9	music_lover	music@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_9.jpg	Music is my escape	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	10	book_worm	books@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_10.jpg	Reading 24/7	2024-12-09 22:25:14
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	Zak	zaki.lbouhmadi@gmail.com	\$2y\$10\$prf7tqyor1g6LX.t8gzAheUKCfRj2mX1xtIIB3xbY...	profile_11.jpg	I am a Java software Engineer	2024-12-16 21:51:44
<input type="checkbox"/>	Éditer	Copier	Supprimer	12	tech_sarah	sarah.tech@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_12.jpg	AI Researcher ML Enthusiast	2024-12-17 01:20:48
<input type="checkbox"/>	Éditer	Copier	Supprimer	13	travel_max	max.adventures@example.com	\$2y\$10\$uiTAAnHHuH3xjhrKetyC.fx2/AwAPXfjZaUx.X9AiX...	profile_13.jpg	Digital Nomad 30 countries	2024-12-17 01:20:48

Console de requêtes SQL

2. **Tweets:** przechowuje tweety, z kluczem obcym do użytkownika, który je utworzył oraz polem tekstowym na zawartość tweeta.

phpMyAdmin

Serveur courant : mysql33.mydevil.net

Récentes Préférées

Nouvelle base de données

m1039_blog

Nouvelle table

comments

follows

likes

tweets

users

Serveur : mysql33.mydevil.net » Base de données : m1039_blog » Table : tweets

Parcourir Structure SQL Rechercher Insérer Exporter Importer Opérations Suivi Déclencheurs

Options supplémentaires

			tweet_id	user_id	content	created_at	image
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	1 Just learned a new programming language! #coding	2024-12-09 22:25:28	tweet_1.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	1 Working on an exciting project! Stay tuned	2024-12-09 22:25:28	tweet_2.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	2 Exploring beautiful Paris today! #travel	2024-12-09 22:25:28	tweet_4.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	2 Best coffee in town! ☕	2024-12-09 22:25:28	tweet_5.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	2 Planning my next adventure! #wanderlust	2024-12-09 22:25:28	tweet_6.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	2 Travel tips coming soon!	2024-12-09 22:25:28	tweet_7.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	3 Debugging this code for hours ? #programming	2024-12-09 22:25:44	tweet_8.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	9	3 New JavaScript framework released! #coding	2024-12-09 22:25:44	tweet_9.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	10	3 Code review time! #developer	2024-12-09 22:25:44	tweet_10.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	3 Built my first AI model! #ML	2024-12-09 22:25:44	tweet_11.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	12	3 Open source is the future	2024-12-09 22:25:44	tweet_12.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	13	4 Captured a beautiful sunset today ?	2024-12-09 22:25:44	tweet_13.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	14	4 New camera gear arrived! #photography	2024-12-09 22:25:44	tweet_14.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	15	4 Photo editing tips thread:	2024-12-09 22:25:44	tweet_15.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	16	5 iPhone 15 review coming soon!	2024-12-09 22:25:44	tweet_16.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	17	5 Top 5 laptops of 2024 #tech	2024-12-09 22:25:44	tweet_17.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	18	5 AI is revolutionizing everything!	2024-12-09 22:25:44	tweet_18.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	19	5 Testing new gadgets today	2024-12-09 22:25:44	tweet_19.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	20	1 snow falling in Wrocław !	2024-12-11 23:37:03	tweet_20.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	21	1 hi there	2024-12-11 23:37:56	tweet_21.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	28	11 tv	2024-12-16 23:44:49	tweet_28.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	29	11 ttt	2024-12-16 23:45:24	tweet_29.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	30	11 sunset	2024-12-17 00:19:36	tweet_30.jpg
<input type="checkbox"/>	Éditer	Copier	Supprimer	31	12 Just deployed my first ML model! #AI	2024-12-17 01:21:40	tweet_31.jpg

3. **Likes:** tabela pośrednia przechowująca polubienia tweetów, z kluczami obcymi do użytkownika i tweeta.

phpMyAdmin

Serveur courant : mysql33.mydevil.net

Récentes Préférées

Nouvelle base de données m1039_blog

Nouvelle table

comments

follows

likes

tweets

users

Parcourir Structure SQL Rechercher Insérer Exporter Importer

user_id tweet_id created_at

<input type="checkbox"/>	Éditer Copier Supprimer	1	2	2024-12-11 23:38:49
<input type="checkbox"/>	Éditer Copier Supprimer	1	14	2024-12-11 23:38:40
<input type="checkbox"/>	Éditer Copier Supprimer	1	15	2024-12-16 21:50:58
<input type="checkbox"/>	Éditer Copier Supprimer	1	21	2024-12-16 20:42:16
<input type="checkbox"/>	Éditer Copier Supprimer	1	28	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	1	29	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	2	1	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	2	28	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	3	1	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	3	29	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	4	5	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	4	30	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	5	4	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	5	28	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	6	5	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	6	29	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	7	8	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	7	30	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	8	7	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	8	28	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	9	10	2024-12-09 22:25:59
<input type="checkbox"/>	Éditer Copier Supprimer	9	29	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	10	30	2024-12-17 01:22:53
<input type="checkbox"/>	Éditer Copier Supprimer	11	8	2024-12-17 00:36:04

4. **Comments:** przechowuje komentarze do tweetów, z kluczami obcymi do użytkownika i tweeta oraz polem tekstowym na zawartość komentarza.

phpMyAdmin

Serveur courant : mysql33.mydevil.net

Récentes Préférées

Nouvelle base de données

m1039_blog

Nouvelle table

comments

follows

likes

tweets

users

Base de données : m1039_blog

Table : comments

comment_id tweet_id user_id content created_at

<input type="checkbox"/>	Éditer	Copier	Supprimer	27	1	2	Which language did you learn? #curious	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	28	1	3	Welcome to the coding world! 🌍	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	29	1	5	Would love to hear your experience!	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	30	4	1	Paris is amazing! Visit the Louvre!	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	31	4	3	Share some pictures! 📸	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	32	5	6	Which coffee shop? I need to try!	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	33	8	1	AI is fascinating! What framework did you use?	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	34	8	2	Would love to collaborate on ML projects	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	35	8	5	Great achievement! 🏆	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	36	13	1	Waiting for the iPhone review!	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	37	13	4	Please cover the camera features	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	38	14	3	Include developer laptops too!	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	39	14	7	What about gaming laptops? 🎮	2024-12-09 22:27:59
<input type="checkbox"/>	Éditer	Copier	Supprimer	40	21	1	nice	2024-12-16 20:42:24
<input type="checkbox"/>	Éditer	Copier	Supprimer	41	28	2	Amazing progress! Keep it up 🙌	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	42	29	3	Love the composition of this shot!	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	43	30	4	Would love to try this recipe	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	44	28	5	Great form! What is your routine?	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	45	29	6	This is inspiring! ✨	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	46	30	7	Could you share more details?	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	47	28	8	Fantastic work! 🌟	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	48	29	9	Thanks for sharing this!	2024-12-17 01:24:10

Tout cocher Avec la sélection : Éditer Copier Supprimer Exporter

Console de requêtes SQL

5. Follows: tabela pośrednia przechowująca relacje obserwowania między użytkownikami.

phpMyAdmin

Serveur courant : mysql33.mydevil.net

Récentes Préférées

Nouvelle base de données

m1039_blog

Nouvelle table

comments

follows

likes

tweets

users

Base de données : m1039_blog

Table : follows

1 > >> | ☐ Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans

Options supplémentaires

<input type="checkbox"/>	Éditer	Copier	Supprimer	1	4	2024-12-16 21:50:37
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	6	2024-12-16 20:42:00
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	8	2024-12-12 00:09:23
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	10	2024-12-11 23:38:12
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	3	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	4	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	5	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	6	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	5	7	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	6	8	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	7	9	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	8	10	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	9	1	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	10	2	2024-12-17 01:24:10
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	1	2024-12-16 23:43:07
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	3	2024-12-16 22:07:27
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	5	2024-12-17 00:58:38
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	7	2024-12-17 00:43:03
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	8	2024-12-16 22:25:55
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	10	2024-12-17 00:15:34
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	12	2024-12-17 02:21:17
<input type="checkbox"/>	Éditer	Copier	Supprimer	11	13	2024-12-17 02:19:37

Console de requêtes SQL

Kod:

```
CREATE TABLE `comments` (  
  `comment_id` int NOT NULL,  
  `tweet_id` int DEFAULT NULL,  
  `user_id` int DEFAULT NULL,  
  `content` mediumtext COLLATE utf8mb4_unicode_ci NOT NULL,  
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
CREATE TABLE `follows` (  
  `follower_id` int NOT NULL,  
  `following_id` int NOT NULL,  
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
CREATE TABLE `likes` (  
  `user_id` int NOT NULL,  
  `tweet_id` int NOT NULL,  
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
CREATE TABLE `tweets` (  
  `tweet_id` int NOT NULL,  
  `user_id` int DEFAULT NULL,  
  `content` mediumtext COLLATE utf8mb4_unicode_ci NOT NULL,  
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,  
  `image` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
CREATE TABLE `users` (  
  `user_id` int NOT NULL,  
  `username` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,  
  `email` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL,  
  `password` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,  
  `profile_pic` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL DEFAULT 'default_profile.jpg',  
  `bio` mediumtext COLLATE utf8mb4_unicode_ci,  
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
ALTER TABLE `comments`  
  ADD PRIMARY KEY (`comment_id`),  
  ADD KEY `tweet_id` (`tweet_id`),  
  ADD KEY `user_id` (`user_id`);
```

```
ALTER TABLE `follows`  
  ADD PRIMARY KEY (`follower_id`, `following_id`),  
  ADD KEY `following_id` (`following_id`);
```

```
ALTER TABLE `likes`  
  ADD PRIMARY KEY (`user_id`, `tweet_id`),  
  ADD KEY `tweet_id` (`tweet_id`);
```

```
ALTER TABLE `tweets`  
  ADD PRIMARY KEY (`tweet_id`),  
  ADD KEY `user_id` (`user_id`);
```

```
ALTER TABLE `users`  
  ADD PRIMARY KEY (`user_id`),  
  ADD UNIQUE KEY `username` (`username`),  
  ADD UNIQUE KEY `email` (`email`);
```

```
ALTER TABLE `comments`  
  MODIFY `comment_id` int NOT NULL AUTO_INCREMENT;
```

```
ALTER TABLE `tweets`  
  MODIFY `tweet_id` int NOT NULL AUTO_INCREMENT;
```

```
ALTER TABLE `users`  
  MODIFY `user_id` int NOT NULL AUTO_INCREMENT;
```

```
ALTER TABLE `comments`  
  ADD CONSTRAINT `comments_ibfk_1` FOREIGN KEY (`tweet_id`) REFERENCES `tweets` (`tweet_id`),  
  ADD CONSTRAINT `comments_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`);
```

```
ALTER TABLE `follows`  
  ADD CONSTRAINT `follows_ibfk_1` FOREIGN KEY (`follower_id`) REFERENCES `users` (`user_id`),  
  ADD CONSTRAINT `follows_ibfk_2` FOREIGN KEY (`following_id`) REFERENCES `users` (`user_id`);
```

```
ALTER TABLE `likes`  
  ADD CONSTRAINT `likes_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`),  
  ADD CONSTRAINT `likes_ibfk_2` FOREIGN KEY (`tweet_id`) REFERENCES `tweets` (`tweet_id`);  
  
ALTER TABLE `tweets`  
  ADD CONSTRAINT `tweets_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`);
```

Tabele zostały zaprojektowane z uwzględnieniem zasad normalizacji bazy danych, aby zminimalizować redundancję i zapewnić integralność danych. Klucze główne i obce utrzymują relacje między tabelami.

W tym rozdziale omówiłem analizę wymagań, architekturę systemu i projekt bazy danych dla mojej aplikacji kłona Twittera. Przełożenie tych założeń na rzeczywistą implementację będzie następnym dużym krokiem w tym projekcie inżynierskim, który będzie głównym tematem kolejnego rozdziału.

Źródła:

[1] <https://www.guru99.com/functional-vs-non-functional-requirements.html>: Functional vs Non Functional Requirements - Guru99 [2] <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>: What is Use Case Diagram? - Visual Paradigm [3] <https://www.altexsoft.com/blog/engineering/what-is-system-design-in-software-engineering/>: What is System Design in Software Engineering? - AltexSoft [4] <https://developer.android.com/topic/architecture>: Guide to app architecture - Android Developers [5] <https://www.mysql.com/products/workbench/>: MySQL Workbench - Visual Database Design [6] https://www.tutorialspoint.com/dbms/database_normalization.htm: Database Normalization - Tutorialspoint

Rozdział 4: Implementacja

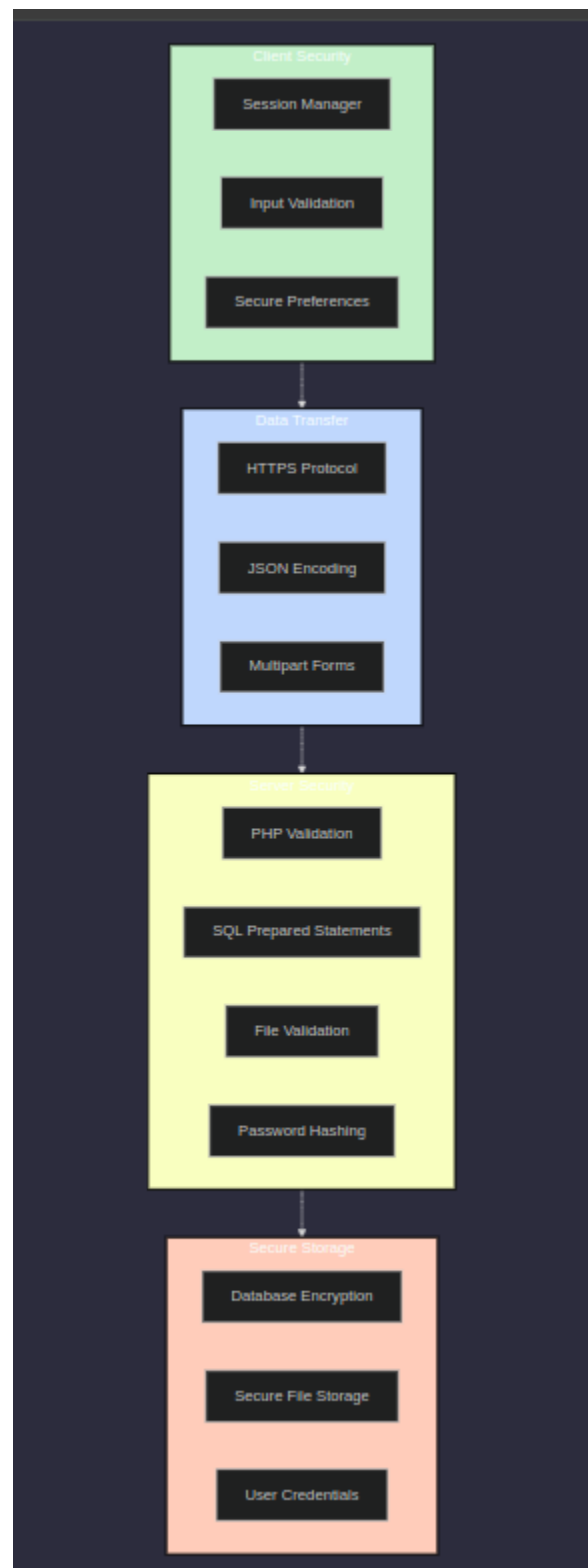
W tym rozdziale omówię szczegóły implementacji mojej aplikacji kłona Twittera na Androida. Podzielę to na części dotyczące architektury bezpieczeństwa, implementacji frontendu (aplikacji Androida), backendu (API PHP) i integracji z bazą danych MySQL.

Architektura bezpieczeństwa systemu

Bezpieczeństwo było jednym z kluczowych aspektów, które wziąłem pod uwagę podczas implementacji mojej aplikacji kłona Twittera. Aby zapewnić, że dane użytkowników są chronione, a system jest odporny na typowe zagrożenia, zaprojektowałem wielowarstwową architekturę bezpieczeństwa.

Jak widać na powyższym diagramie, architektura bezpieczeństwa mojej aplikacji składa się z kilku kluczowych komponentów:

1. Szyfrowanie bazy danych: Wszystkie wrażliwe dane, takie jak hasła użytkowników, są szyfrowane przed zapisaniem w bazie danych MySQL. Zapewnia to dodatkową warstwę ochrony w przypadku naruszenia bezpieczeństwa bazy danych.
2. Bezpieczne przechowywanie haseł: Hasła użytkowników nigdy nie są przechowywane w postaci plaintext. Zamiast tego, używam algorytmu haszowania (np. bcrypt) do generowania i przechowywania hashy haseł. Podczas uwierzytelniania, podane hasło jest haszowane i porównywane z przechowywanymi hashami.
3. Walidacja po stronie serwera: Wszystkie dane wejściowe od użytkownika, takie jak dane logowania czy rejestracji, są dokładnie sprawdzane i walidowane po stronie serwera (w skryptach PHP). Pomaga to zapobiegać atakom typu SQL Injection czy wprowadzaniu nieprawidłowych danych do systemu.
4. Sesje i bezpieczne uwierzytelnianie: Po pomyślnym zalogowaniu, tworzony jest unikalny token sesji dla użytkownika. Token ten jest następnie przesyłany przy każdym kolejnym żądaniu i weryfikowany po stronie serwera. Pomaga to utrzymać stan uwierzytelnienia użytkownika w bezpieczny sposób.



Implementacja tych mechanizmów bezpieczeństwa wymagała starannego planowania i testowania. Zwłaszcza poprawne szyfrowanie i przechowywanie haseł okazało się być tricky,

ponieważ wymagało zrozumienia różnych algorytmów haszowania i ich odpowiedniego wykorzystania w PHP.

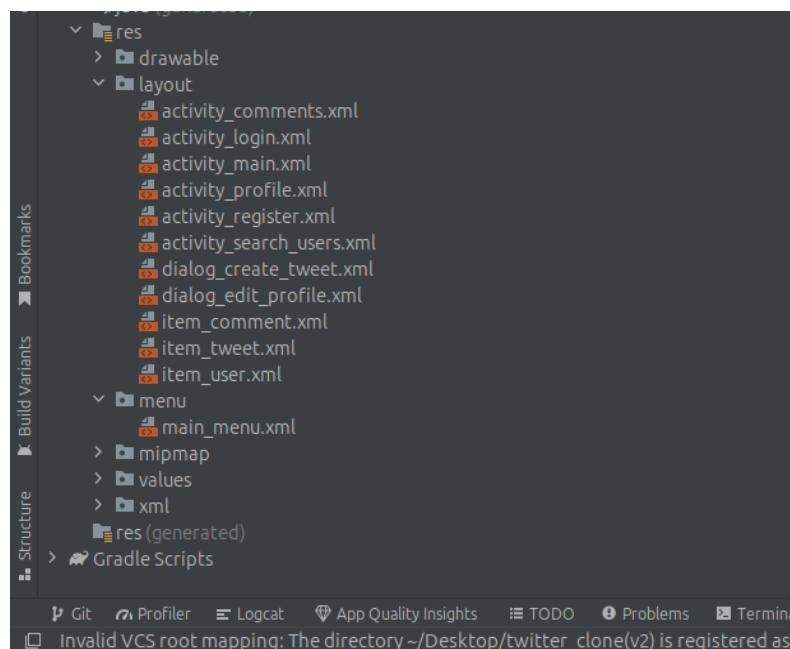
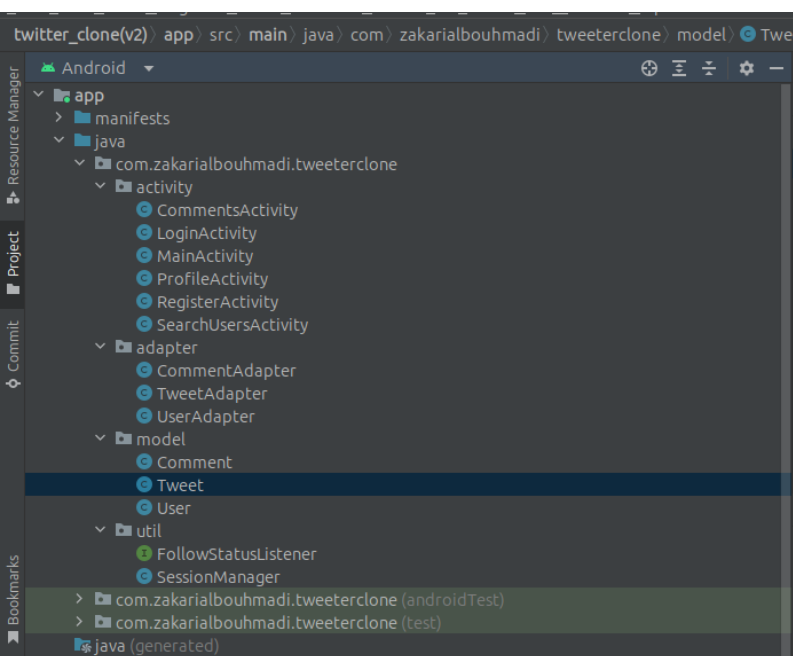
Mimo tych wyzwań, jestem zadowolony z osiągniętego wyniku. System, chociaż nie jest doskonały, zapewnia solidne podstawy bezpieczeństwa, chroniąc dane użytkowników i minimalizując ryzyko typowych ataków.

Oczywiście, jest jeszcze wiele miejsca na ulepszenia. W przyszłości, chciałbym zbadać dodatkowe środki bezpieczeństwa, takie jak dwuskładnikowe uwierzytelnianie czy bardziej rygorystyczne zasady dotyczące haseł. Niemniej jednak, ta implementacja dała mi cenne doświadczenie w projektowaniu i wdrażaniu mechanizmów bezpieczeństwa w aplikacjach internetowych.

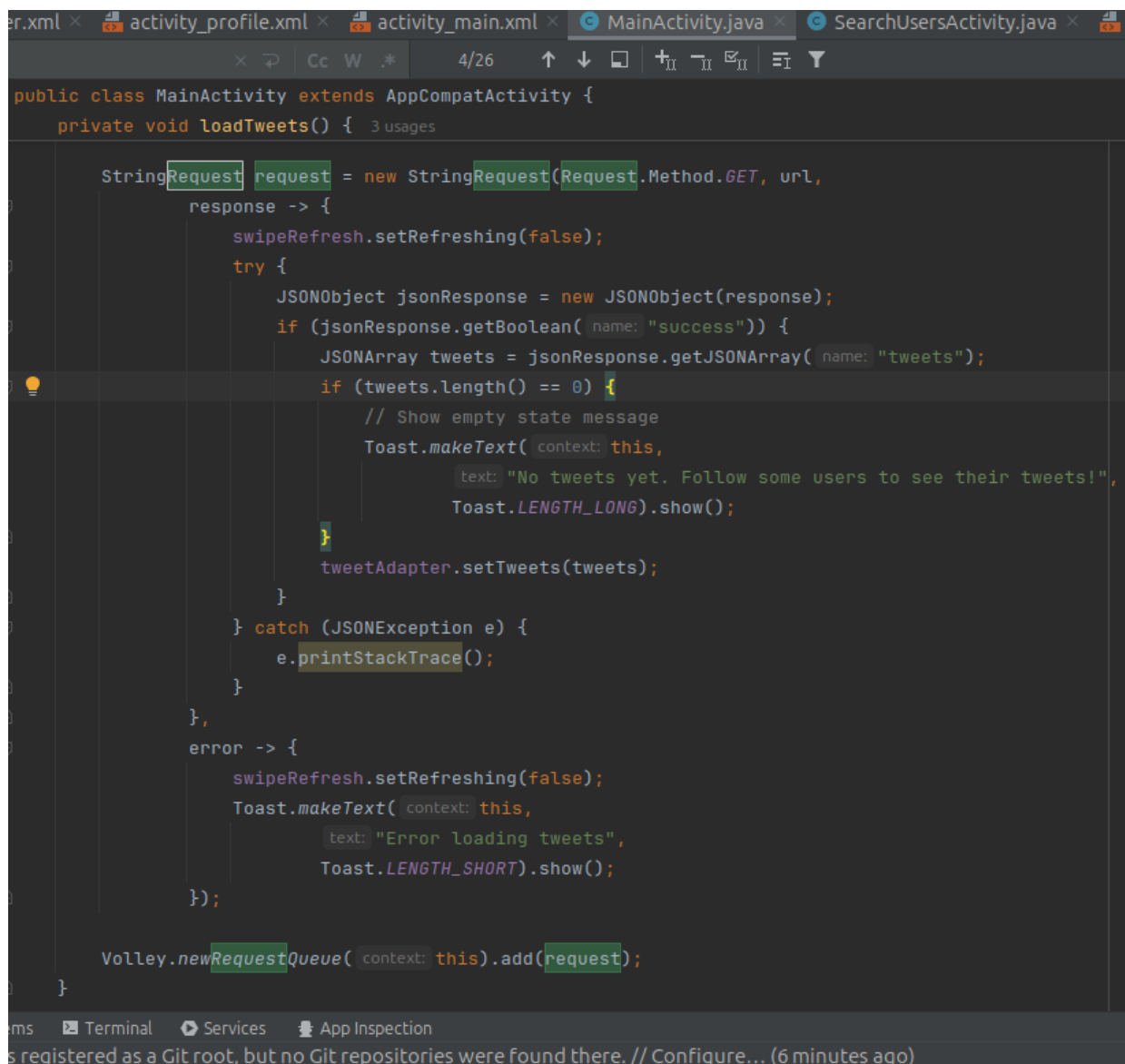
Implementacja Frontendu (Aplikacja Androida)

Aplikacja Androida została zaimplementowana w języku Java przy użyciu Android Studio jako głównego środowiska IDE. Projekt jest zorganizowany zgodnie ze standardowymi praktykami, z oddzielnymi pakietami dla activities, adapters i models.

Główne ekrany aplikacji, takie jak **LoginActivity**, **RegisterActivity**, **MainActivity** (dla strony głównej), **ProfileActivity**, zostały zaimplementowane jako Activities. Adaptery, takie jak **TweetAdapter** i **CommentAdapter**, są odpowiedzialne za wyświetlanie list tweetów i komentarzy w **RecyclerViews**.



Do komunikacji z serwerem użyłem biblioteki **Volley** - popularnej biblioteki sieciowej dla Androida. Zapytania sieciowe, takie jak logowanie, rejestracja, pobieranie tweetów itp., są wykonywane asynchronicznie za pomocą `StringRequest` lub `JsonObjectRequest`.



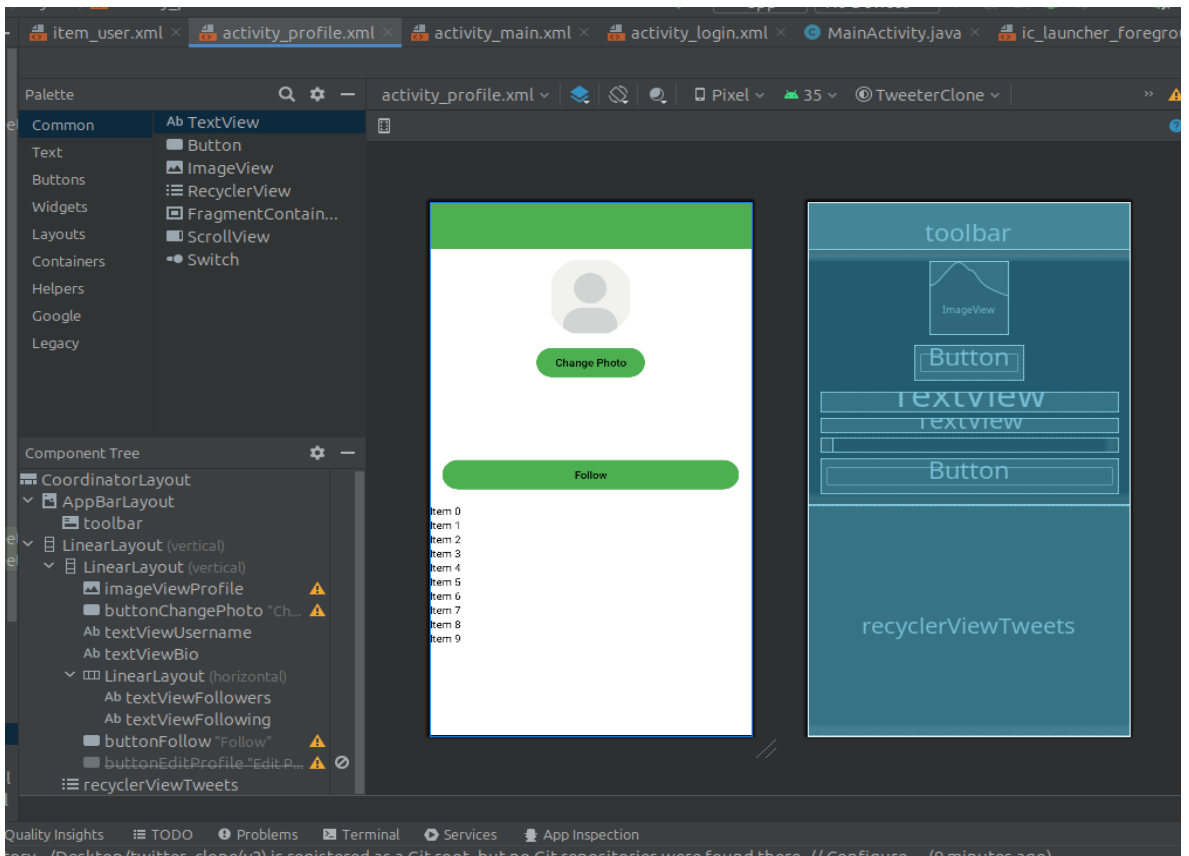
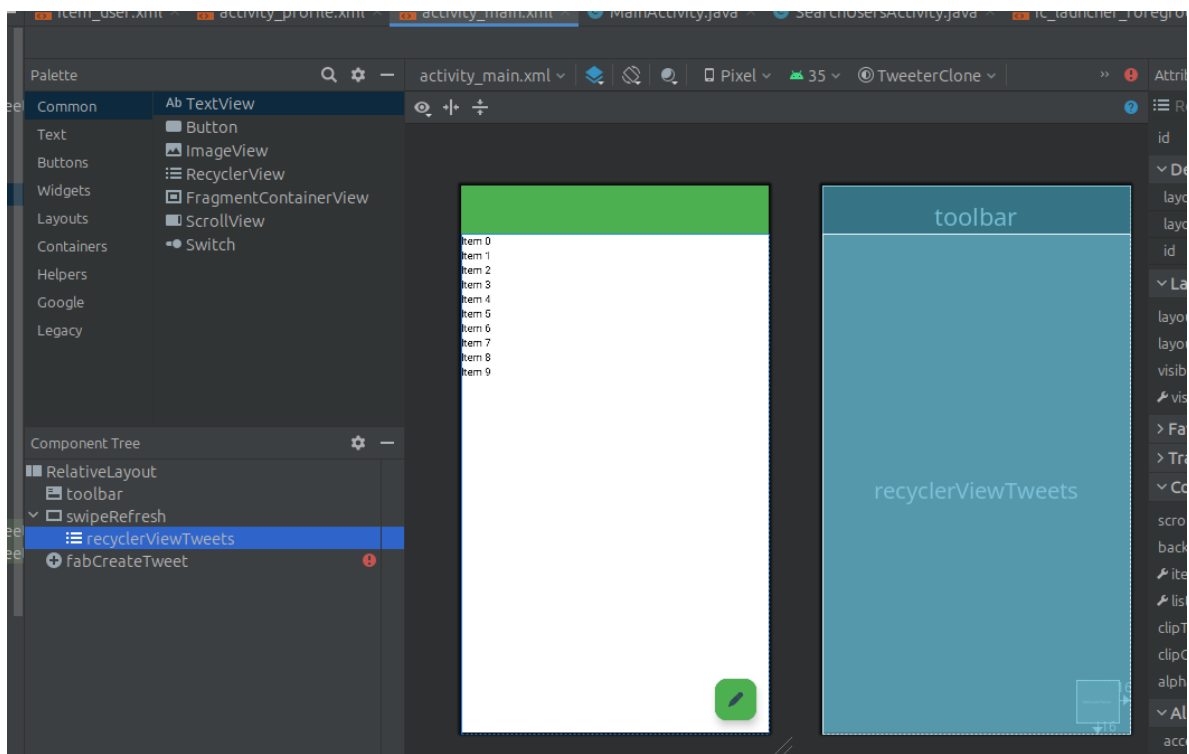
```
public class MainActivity extends AppCompatActivity {
    private void loadTweets() { 3 usages

        StringRequest request = new StringRequest(Request.Method.GET, url,
            response -> {
                swipeRefresh.setRefreshing(false);
                try {
                    JSONObject jsonResponse = new JSONObject(response);
                    if (jsonResponse.getBoolean(name: "success")) {
                        JSONArray tweets = jsonResponse.getJSONArray(name: "tweets");
                        if (tweets.length() == 0) {
                            // Show empty state message
                            Toast.makeText(context: this,
                                text: "No tweets yet. Follow some users to see their tweets!",
                                Toast.LENGTH_LONG).show();
                        }
                        tweetAdapter.setTweets(tweets);
                    }
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            },
            error -> {
                swipeRefresh.setRefreshing(false);
                Toast.makeText(context: this,
                    text: "Error loading tweets",
                    Toast.LENGTH_SHORT).show();
            });

        Volley.newRequestQueue(context: this).add(request);
    }
}
```

Do przechowywania informacji o sesji użytkownika, takich jak token uwierzytelniający i ID użytkownika, użyłem **SharedPreferences**. Tutaj napotkałem pewne wyzwania w zabezpieczeniu tych danych i ostatecznie zdecydowałem się na proste rozwiązanie przechowywania ich w trybie prywatnym.

Implementacja obsługi różnych rozmiarów ekranów i responsywnego UI była kolejnym wyzwaniem. Wykorzystałem **RelativeLayout** i **LinearLayout**, aby elementy układały się poprawnie na różnych ekranach.

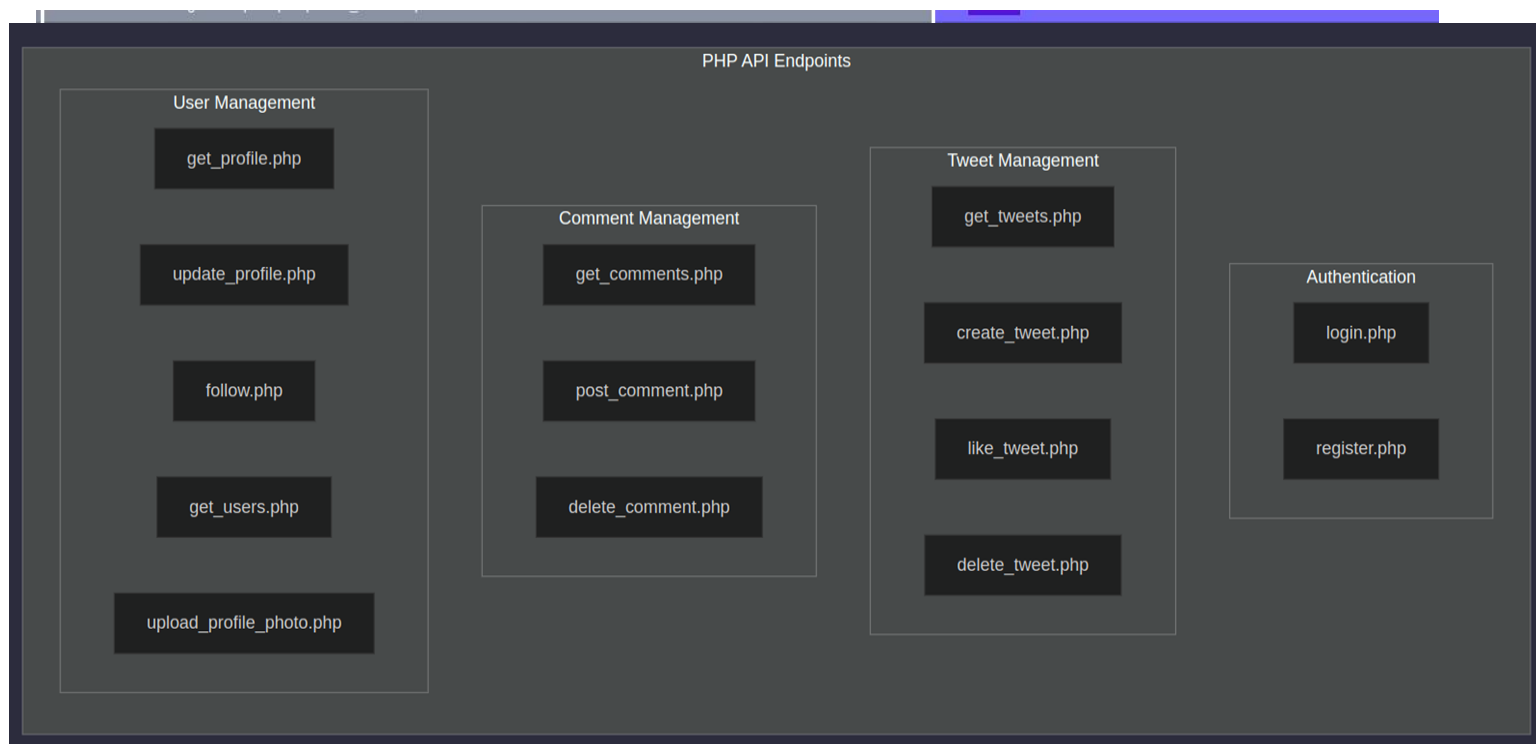


Do obsługi ładowania obrazów, takich jak zdjęcie profilowe użytkownika, użyłem biblioteki Glide. Zapewnia ona wydajne ładowanie i cachowanie obrazów z URL.

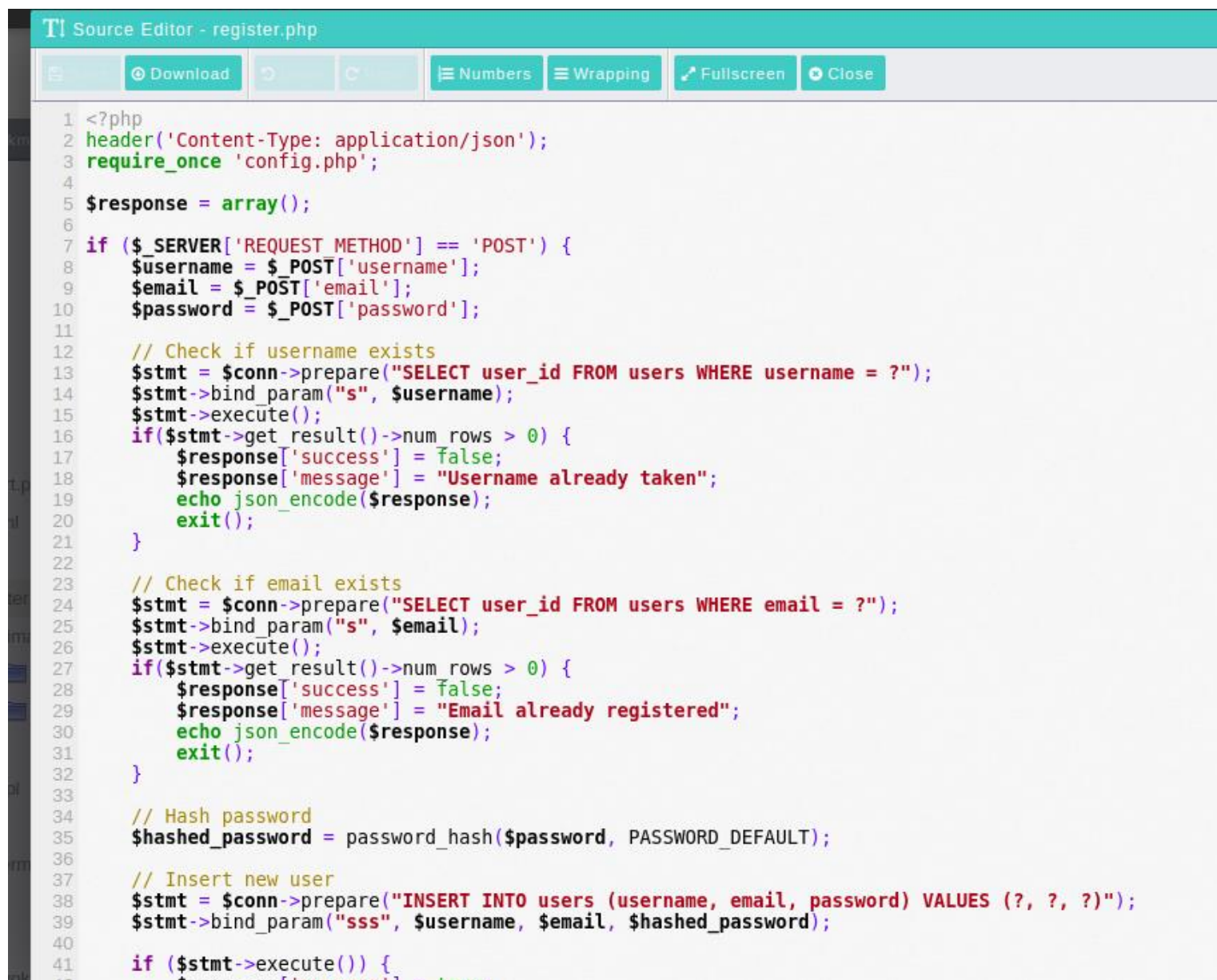
```
Log.d( tag: "ProfileUpload", msg: "Image URL received: " + im
Glide.with( activity: this) RequestManager
    .load(imageUrl) RequestBuilder<Drawable>
    .placeholder(R.drawable.ic_launcher_foreground)
    .into(imageViewProfile);
Toast.makeText( context: this, text: "Profile photo updated", 1
use f
```

Implementacja Backendu (API PHP)

Backend aplikacji został zaimplementowany w PHP jako zestaw endpointów API. Każdy endpoint jest odpowiedzialny za konkretne zadanie, takie jak rejestracja użytkownika, logowanie, pobieranie tweetów itp. Dane są zwracane w formacie JSON do przetworzenia przez aplikację Androida.



Użyłem wbudowanego w PHP wsparcia dla MySQL do interakcji z bazą danych. Zapytania są wykonywane przy użyciu prepared statements w celu zapobiegania atakom SQL injection. Oto przykład endpointu rejestracji użytkownika:



```
T! Source Editor - register.php
Download
Numbers
Wrapping
Fullscreen
Close

1 <?php
2 header('Content-Type: application/json');
3 require_once 'config.php';
4
5 $response = array();
6
7 if ($_SERVER['REQUEST_METHOD'] == 'POST') {
8     $username = $_POST['username'];
9     $email = $_POST['email'];
10    $password = $_POST['password'];
11
12    // Check if username exists
13    $stmt = $conn->prepare("SELECT user_id FROM users WHERE username = ?");
14    $stmt->bind_param("s", $username);
15    $stmt->execute();
16    if($stmt->get_result()->num rows > 0) {
17        $response['success'] = false;
18        $response['message'] = "Username already taken";
19        echo json_encode($response);
20        exit();
21    }
22
23    // Check if email exists
24    $stmt = $conn->prepare("SELECT user_id FROM users WHERE email = ?");
25    $stmt->bind_param("s", $email);
26    $stmt->execute();
27    if($stmt->get_result()->num rows > 0) {
28        $response['success'] = false;
29        $response['message'] = "Email already registered";
30        echo json_encode($response);
31        exit();
32    }
33
34    // Hash password
35    $hashed_password = password_hash($password, PASSWORD_DEFAULT);
36
37    // Insert new user
38    $stmt = $conn->prepare("INSERT INTO users (username, email, password) VALUES (?, ?, ?)");
39    $stmt->bind_param("sss", $username, $email, $hashed_password);
40
41    if ($stmt->execute()) {
42        $response['success'] = true;
```

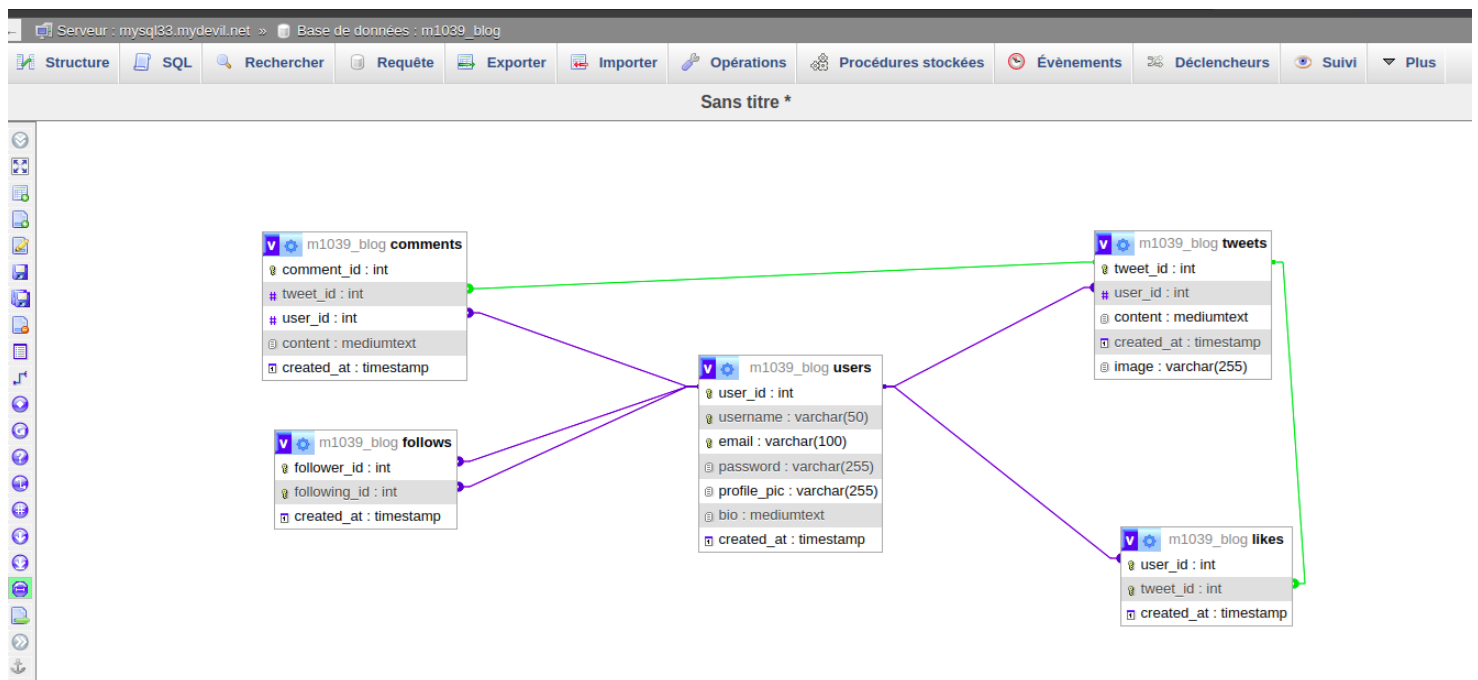
```

40
41     if ($stmt->execute()) {
42         $response['success'] = true;
43         $response['message'] = "Registration successful";
44     } else {
45         $response['success'] = false;
46         $response['message'] = "Registration failed";
47     }
48 } else {
49     $response['success'] = false;
50     $response['message'] = "Invalid request method";
51 }
52
53 echo json_encode($response);
54 ?>

```

Integracja z Bazą Danych MySQL

Baza danych MySQL jest hostowana online na MyDevil - polskim dostawcy hostingowym. Zaimplementowałem bazę danych zgodnie ze schematem zaprojektowanym w rozdziale 3.



Do interakcji z bazą danych z poziomu PHP używam biblioteki MySQLi. Połączenie z bazą danych jest nawiązywane przy użyciu danych uwierzytelniających przechowywanych w oddzielnym pliku konfiguracyjnym, który nie jest śledzony przez system kontroli wersji ze względów bezpieczeństwa.

W procesie integracji napotkałem problemy związane z kodowaniem znaków i obsługą polskich znaków diakrytycznych. Rozwiązałem to przez ustawienie kodowania połączenia na UTF-8. Kolejnym wyzwaniem było efektywne pobieranie i łączenie danych z wielu powiązanych tabel. Użyłem złączeń (JOINS) i podkwerend, aby zoptymalizować zapytania i zminimalizować liczbę oddzielnych zapytań do bazy danych.


```
Source Editor - get_profile.php
Download
Numbers
Wrapping
Fullscreen
Close

33     throw new Exception("User not found");
34 }
35
36 $profile = $profile_result->fetch_assoc();
37
38 // Get user's tweets
39 $tweets_query = "
40     SELECT
41         t.*,
42         u.username,
43         (SELECT COUNT(*) FROM likes WHERE tweet_id = t.tweet_id) as likes_count,
44         (SELECT COUNT(*) FROM comments WHERE tweet_id = t.tweet_id) as comments_count,
45         EXISTS(SELECT 1 FROM likes WHERE tweet_id = t.tweet_id AND user_id = ?) as is_liked
46     FROM tweets t
47     JOIN users u ON t.user_id = u.user_id
48     WHERE t.user_id = ?
49     ORDER BY t.created_at DESC
50 ";
51
52 $stmt = $conn->prepare($tweets_query);
53 $stmt->bind_param("ii", $current_user_id, $user_id);
54 $stmt->execute();
55 $tweets_result = $stmt->get_result();
56
57 $tweets = array();
58 while ($row = $tweets_result->fetch_assoc()) {
59     $tweets[] = array(
60         'tweet id' => (int)$row['tweet_id'],
61         'user id' => (int)$row['user_id'],
62         'username' => $row['username'],
63         'content' => $row['content'],
64         'created at' => $row['created_at'],
65         'likes count' => (int)$row['likes_count'],
66         'comments count' => (int)$row['comments_count'],
67         'is liked' => (bool)$row['is_liked']
68     );
69 }
70
71 $response = array(
72     'success' => true,
73     'profile' => array(
74         'user id' => (int)$profile['user_id'],
```

W tym rozdziale szczegółowo omówiłem implementację frontendu, backendu i integracji z bazą danych mojej aplikacji kłona Twittera. W procesie tym napotkałem różne wyzwania, ale poprzez badania, debugowanie i eksperymentowanie udało mi się znaleźć satysfakcjonujące rozwiązania. W kolejnym rozdziale opiszę proces testowania aplikacji i przedstawię wyniki.

Źródła:

[1] <https://developer.android.com/studio/intro>: Meet Android Studio - Android Developers

[2] <https://developer.android.com/guide/components/activities/intro-activities>: Introduction to Activities - Android Developers

- [3] <https://developer.android.com/training/volley>: Transmitting Network Data Using Volley - Android Developers
- [4] <https://developer.android.com/training/data-storage/shared-preferences>: Save key-value data - Android Developers
- [5] <https://square.github.io/okhttp/>: OkHttp
- [6] <https://bumptech.github.io/glide/>: Glide
- [7] <https://www.php.net/manual/en/book.mysql.php>: MySQL Improved Extension - PHP.net
- [8] <https://jwt.io/introduction>: Introduction to JSON Web Tokens – JWT.IO

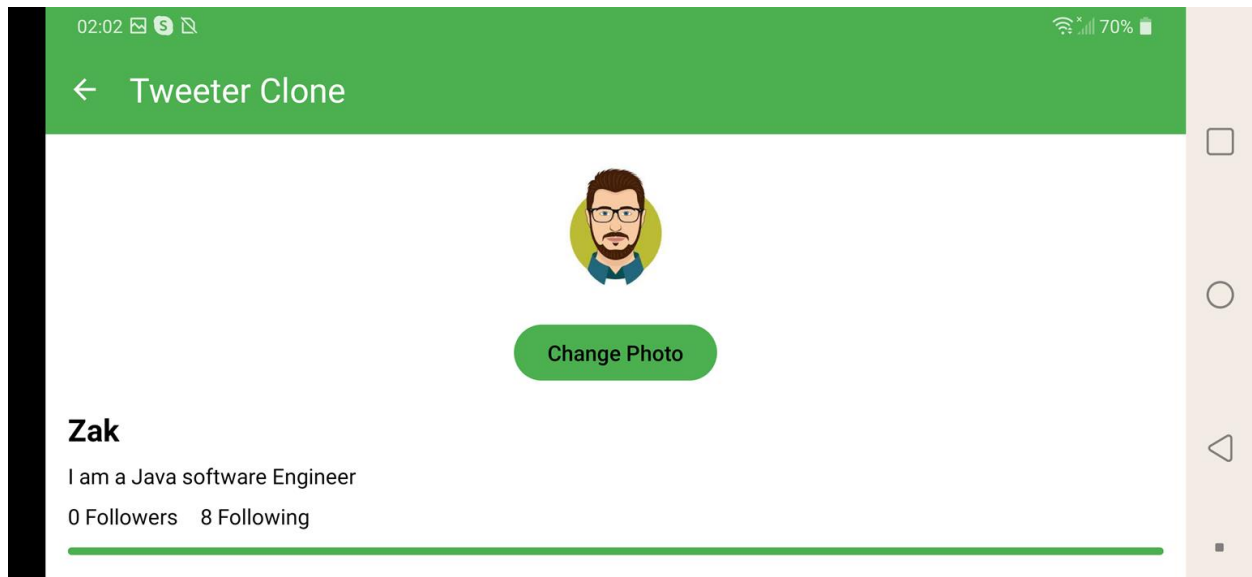
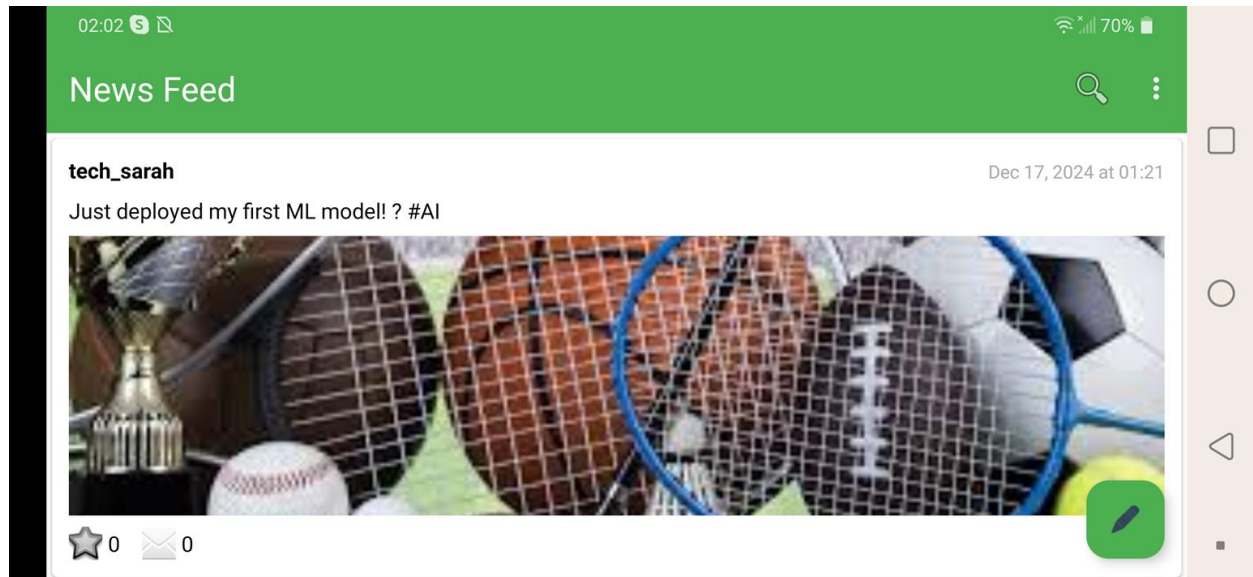
Rozdział 5: Testowanie i Wyniki

W tym rozdziale opiszę proces testowania mojej aplikacji klona Twittera, zarówno z perspektywy frontendu (aplikacji Androida), jak i backendu (API PHP). Przedstawię też ostateczne rezultaty projektu.

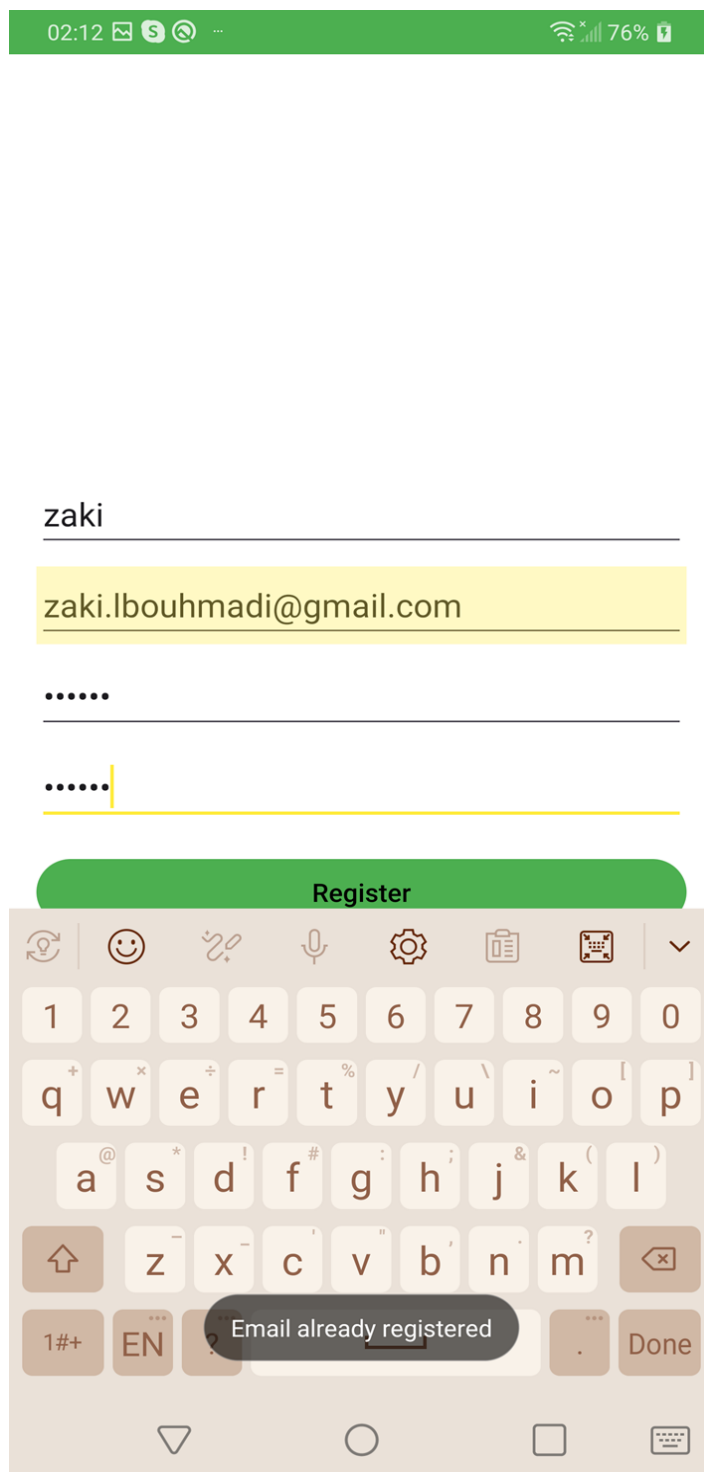
Testowanie Frontendu

Testowanie aplikacji Androida było prowadzone w dwóch głównych formach: testowanie manualne i testowanie automatyczne.

W ramach testowania manualnego, aplikacja była instalowana i testowana na różnych fizycznych urządzeniach z różnymi wersjami Androida, rozmiarami ekranu i rozdzielczościami. Pozwoliło to na sprawdzenie responsywności UI, poprawności działania na różnych wersjach systemu i wydajności na urządzeniach o różnej mocy obliczeniowej. W trakcie tego procesu zidentyfikowano i naprawiono kilka problemów związanych z układem UI i kompatybilnością wsteczną.



Prowadziłem też manualne testowanie wszystkich kluczowych scenariuszy użycia, takich jak rejestracja użytkownika, logowanie, tworzenie tweetów, polubienia, komentarze itp. Scenariusze te były testowane zarówno dla przypadków pozytywnych (np. poprawne dane logowania), jak i negatywnych (np. próba rejestracji z już istniejącym adresem e-mail). Pomogło to wykryć i naprawić kilka błędów logiki aplikacji i obsługi błędów.



Testowanie Backendu

Testowanie backendu skupiało się głównie na testowaniu endpointów API i ich interakcji z bazą danych. Używałem Postman - popularnego narzędzia do testowania API - do wysyłania żądań do endpointów i weryfikowania poprawności odpowiedzi.

Każdy endpoint API był testowany pod kątem różnych scenariuszy wejściowych, w tym poprawnych żądań, brakujących parametrów, nieprawidłowego formatu danych itp. Odpowiedzi były sprawdzane pod kątem poprawnej struktury JSON, kodów statusu HTTP i aktualizacji bazy danych.

Testy obejmowały też scenariusze uwierzytelniania i autoryzacji, aby upewnić się, że tylko uwierzytelnieni użytkownicy mają dostęp do chronionych zasobów, a niewierzytelnione żądania są prawidłowo odrzucane.

Kod:

```
// Register Tests
```

```
pm.test("Register new user", function () {  
    const jsonData = pm.response.json();  
    pm.expect(jsonData.success).to.be.true;  
    pm.expect(jsonData.message).to.include("Registration successful");  
});
```

```
pm.test("Register with existing username", function () {  
    const jsonData = pm.response.json();  
    pm.expect(jsonData.success).to.be.false;  
    pm.expect(jsonData.message).to.equal("Username already taken");  
});
```

```
// Login Tests
```

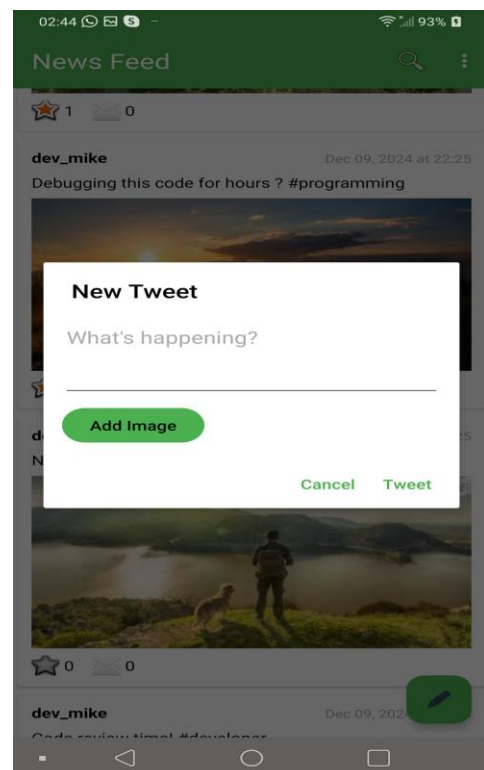
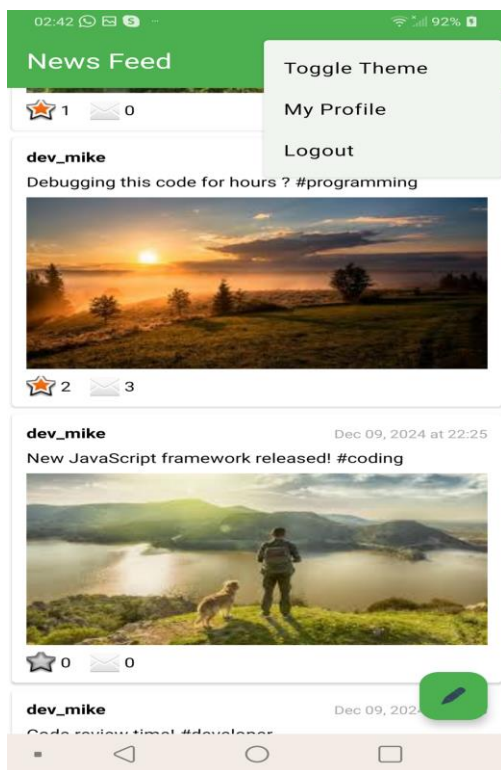
```
pm.test("Successful login", function () {  
    const jsonData = pm.response.json();  
    pm.expect(jsonData.success).to.be.true;  
    pm.expect(jsonData).to.have.property('user_id');  
    pm.expect(jsonData).to.have.property('username');
```

```
// Save tokens for subsequent requests
if(jsonData.user_id) {
    pm.environment.set("user_id", jsonData.user_id);
}
});

pm.test("Login with wrong password", function () {
    const jsonData = pm.response.json();
    pm.expect(jsonData.success).to.be.false;
    pm.expect(jsonData.message).to.equal("Invalid password");
});
```

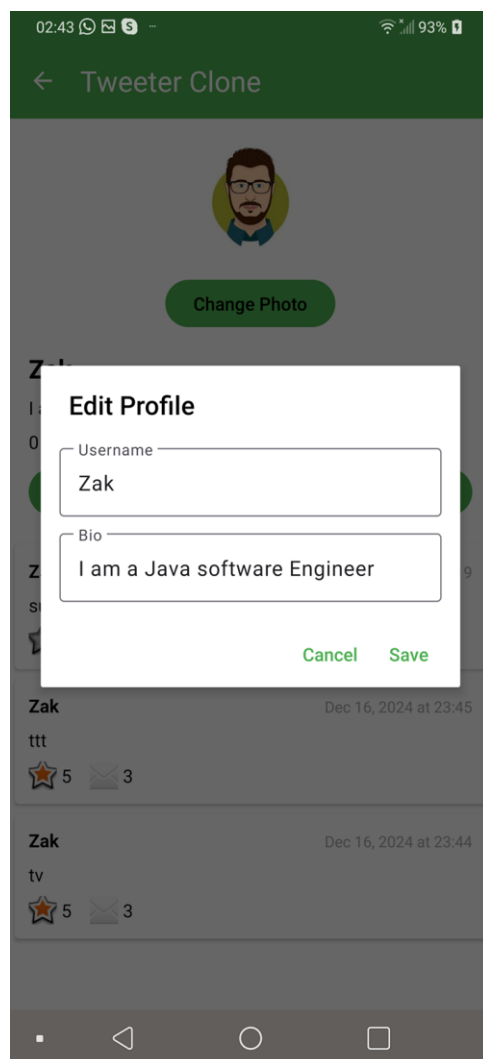
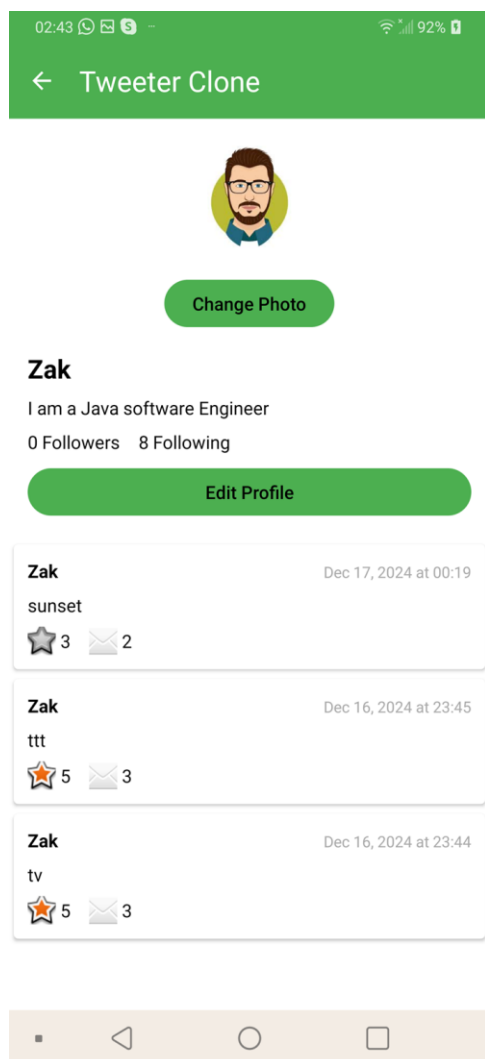
Wyniki i Prezentacja Aplikacji

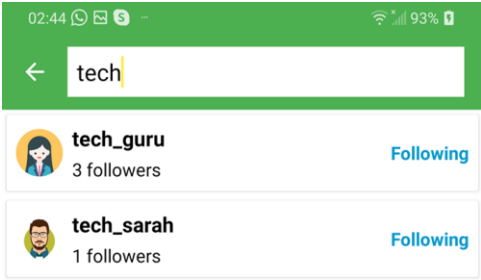
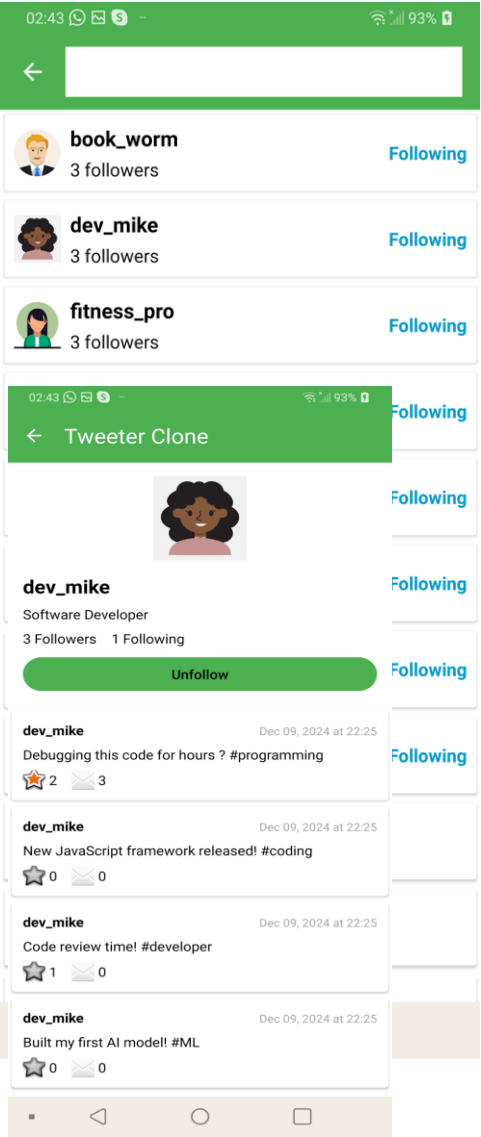
Po wielu iteracjach implementacji i testowania, ostateczna wersja aplikacji spełnia większość początkowych wymagań i oferuje podstawowy zestaw funkcji mediów społecznościowych.



Użytkownicy mogą tworzyć konta, logować się, publikować tweety, przeglądać tweety innych użytkowników, obserwować innych

użytkowników i wchodzić z nimi w interakcje poprzez polubienia i komentarze. Każdy użytkownik ma swój profil pokazujący jego aktywność i statystyki.

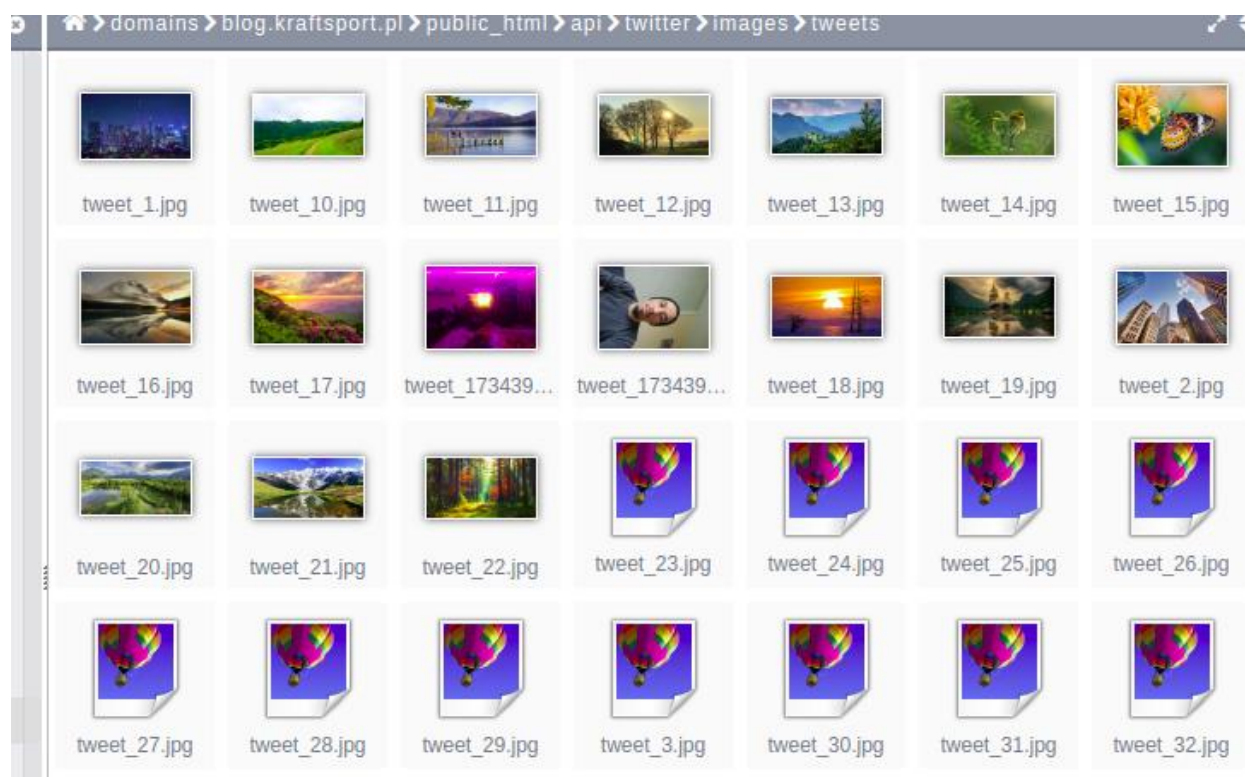




Backend skutecznie obsługuje różne żądania z aplikacji i bezpiecznie przechowuje dane w bazie danych MySQL. Uwierzytelnianie użytkowników i autoryzacja działają zgodnie z założeniami.



profile_pictures_screenshot



posts_pictures_screenshot

Jednak aplikacja ma kilka ograniczeń i obszarów do poprawy. Na przykład, nie ma jeszcze funkcji edycji tweetów. Nie ma też systemu powiadomień w czasie rzeczywistym o nowych interakcjach. UI, choć funkcjonalny, mógłby skorzystać z dalszych ulepszeń i dostosowań w zakresie użyteczności i estetyki wizualnej.

Podsumowując, ten projekt był niezwykle pouczającym doświadczeniem, które pozwoliło mi zastosować w praktyce i poszerzyć moją wiedzę z zakresu programowania aplikacji mobilnych, tworzenia interfejsów API i integracji z bazą danych. Chociaż efekt końcowy nie jest idealny, jestem dumny z tego, co udało mi się osiągnąć i z narzędzia, które mogę teraz używać do prywatnej komunikacji z rodziną i przyjaciółmi.

W tym rozdziale podsumowałem proces testowania aplikacji, zaprezentowałem jej kluczowe funkcje i omówiłem obszary do potencjalnych ulepszeń. W następnym, ostatnim rozdziale, podzielę się końcowymi przemyśleniami i omówię możliwe kierunki przyszłego rozwoju projektu.

Źródła:

- [1] <https://developer.android.com/training/testing/fundamentals>: Android Testing Fundamentals - Android Developers
- [2] <https://www.guru99.com/manual-testing.html>: Manual Testing Tutorial: What is, Concepts, Types & Tool - Guru99
- [3] <https://junit.org/junit5/docs/current/user-guide/>: JUnit 5 User Guide
- [4] <https://www.postman.com/>: Postman API Platform
- [5] <https://owasp.org/www-project-web-security-testing-guide/v42/>: OWASP Web Security Testing Guide
- [6] <https://developer.android.com/docs/quality-guidelines/core-app-quality>: Core app quality - Android Developers

Rozdział 6: Wnioski i Przyszłe Prace

W tej pracy inżynierskiej przedstawiłem proces projektowania i tworzenia aplikacji kłona Twittera na Androida od podstaw. Projekt obejmował analizę wymagań, projektowanie architektury i bazy danych, implementację frontendu i backendu oraz gruntowne testowanie.

Jednym z głównych wniosków z tego projektu jest to, że tworzenie aplikacji mobilnych z backendem jest złożonym, wielowarstwowym procesem, który wymaga różnorodnego zestawu umiejętności i technologii. Skuteczna komunikacja i transfer danych między aplikacją kliencką a serwerem to kluczowe aspekty, które wymagają starannego planowania i implementacji.

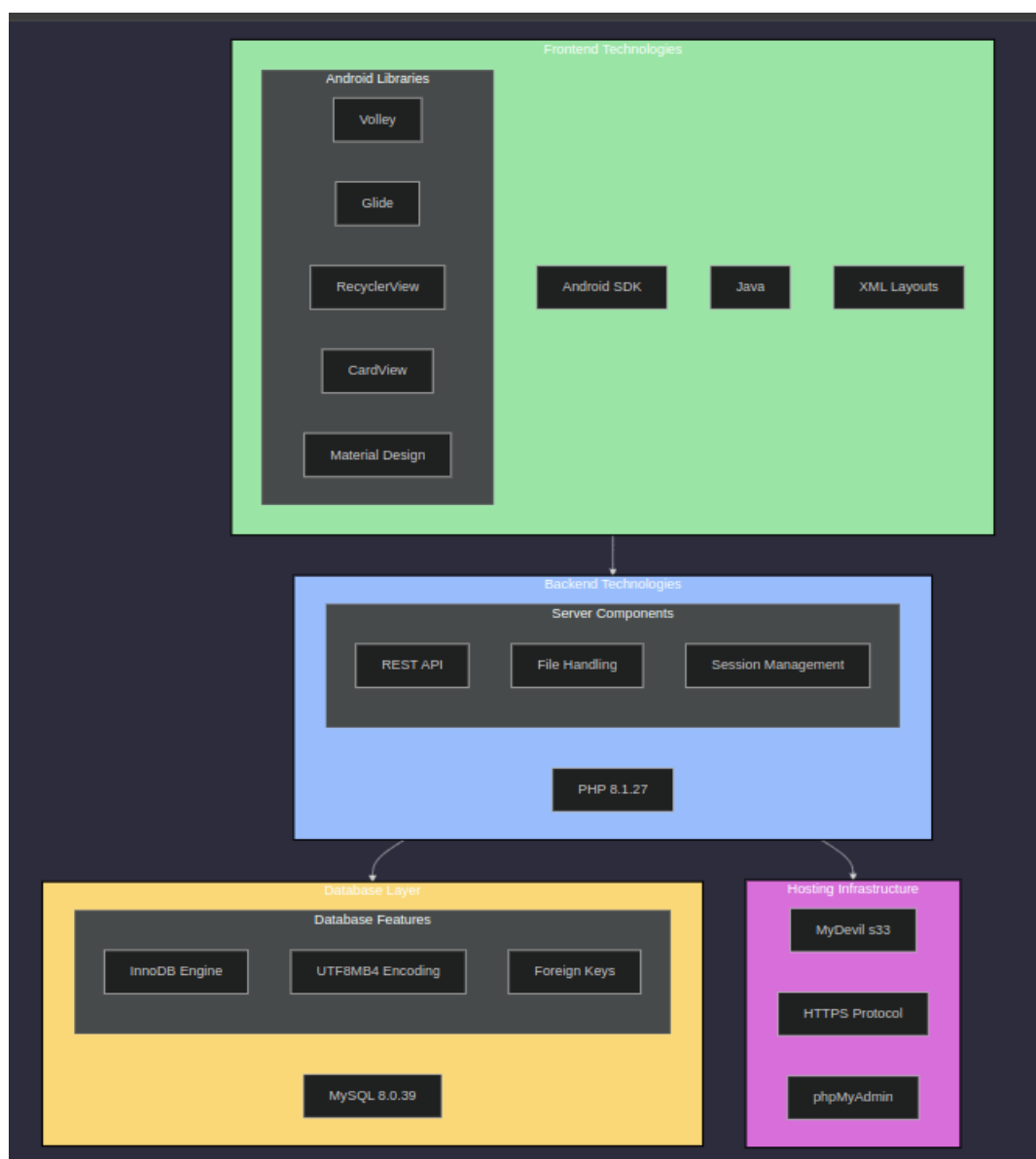


complete_architecture_diagram

Kolejną kluczową refleksją jest znaczenie testowania na każdym etapie procesu tworzenia oprogramowania. Regularne testowanie, zarówno manualne, jak i automatyczne, pomaga wykryć i naprawić błędy wcześniej, zanim staną się one większymi problemami. Jest to szczególnie ważne w przypadku aplikacji mobilnych, które muszą działać na różnych urządzeniach i wersjach systemu.

Z technicznego punktu widzenia, ten projekt dał mi praktyczne doświadczenie w używaniu Javy i frameworka Android do tworzenia aplikacji mobilnych, PHP do tworzenia interfejsów API i

MySQL do persystencji danych. Zdobyłem cenne doświadczenie w integracji tych technologii w celu stworzenia funkcjonalnego systemu end-to-end.



technology_diagram

Jeśli chodzi o potencjalne przyszłe prace i ulepszenia projektu, jest kilka kluczowych obszarów do rozważenia:

1. Rozszerzenie funkcjonalności: Dodanie funkcji takich jak edycja i usuwanie tweetów, system powiadomień w czasie rzeczywistym, prywatne wiadomości między użytkownikami itp.
2. Ulepszenie UI/UX: Przeprojektowanie niektórych elementów interfejsu użytkownika w celu poprawy użyteczności i wizualnej atrakcyjności aplikacji.

3. Zwiększenie bezpieczeństwa: Wdrożenie silniejszych środków bezpieczeństwa, takich jak szyfrowanie haseł, ochrona przed atakami typu denial-of-service, bezpieczniejsze przechowywanie wrażliwych danych itp.
4. Skalowalność: Zoptymalizowanie aplikacji i infrastruktury backendowej pod kątem obsługi większej liczby użytkowników i obciążenia.
5. Testy automatyczne: Rozszerzenie zestawu testów automatycznych, szczególnie o testy UI, aby szybciej wykrywać regresje i zapewnić stabilność aplikacji.

Podsumowując, ten projekt był niezwykle satysfakcjonującym i pouczającym doświadczeniem, które znacząco poszerzyło moje umiejętności w zakresie rozwoju aplikacji mobilnych i web developmentu. Stworzona aplikacja, choć nie jest idealna, spełnia swój podstawowy cel i stanowi solidną podstawę do dalszych ulepszeń i rozbudowy.

Patrząc w przyszłość, postrzegam ten projekt nie jako koniec, ale raczej jako początek mojej podróży w świat rozwoju oprogramowania. Umiejętności i doświadczenia zdobyte podczas tej pracy inżynierskiej z pewnością będą kluczowe w moich przyszłych przedsięwzięciach programistycznych, zarówno akademickich, jak i profesjonalnych.

Z niecierpliwością czekam na wdrożenie aplikacji i udostępnienie jej mojej rodzinie i przyjaciołom. Ich opinie i doświadczenia użytkowników będą nieocenione w kierowaniu przyszłym rozwojem i ulepszaniem aplikacji.

Na koniec chciałbym podziękować mojemu promotorowi i wszystkim, którzy wsparli mnie w tym projekcie. Wasze wskazówki, zachęty i wsparcie były nieocenione na każdym etapie tego przedsięwzięcia.

Z poważaniem,

Zakaria Lbouhmadi 86173