**DODATKOWE ELEMENTY PODCZAS CAŁEGO ŻYCIA APLIKACJI:**

- dalsze poprawki struktury kodu + ewentualne ujednolicenia ikonek, tekstów, styli, błędy z konsoli, refactoring kodu oraz tłumaczenia brakujących tekstów

- pamiętać o commitach i co jakiś czas backupy

**EKRAN GŁÓWNY GRY i EKRAN KARTY - poprawki:**

//przemyślenie mechaniki do końca – wygląd, układ, reponsywność

TODO:  
- aplikacja gdy nie ma internetu się nie uruchamiała – wywalała się na firebase messaging – problem był jakiś z tym, ze nie mogło się zainicjować – trzeba to sprawdzić ocb //OK, trzeba sprawdzić czy w trakcie działania jak wroci internet czy zadziała – to chyba nie zadzialalo tylo za 1 raze mgdy było cache wyczyszczone   
  
- też zauważyłem, że był problem na emulatorze i w sumie na telefonie, że przez pewien czas nie dało się przejść z reklamy interstitial do gry – po kilku próbach to przeszło ale dlaczego? – czy to był lag ze względu na zmianę urządzenia/dysku itp. czy przez to, że są to próbne reklamy i się sypało? Nie czaje za bardzo – pewnie wyjdzie jak wyślę apke do sprawdzenia? – trzebaby potestować wersję deployową na jakimś innym telu ewentualnie  
  
- na ekranie wyboru drużyny trzeba zrobić aby kliknięcie w kolory robiło to samo co przycisk zmieniający kolory ew. dodać jakiegoś hinta „?” opisującego co możesz tutaj zrobić  
  
- nadal jest problem z zatrzymywaniem i odtwarzaniem dźwięku, ten provider nie do końca działa poprawnie, raz się wyłącza ten dźwięk poprzez setting-provider i settings.persistence a raz nie, teraz konieczne jest to aby dźwięk wyłączyć dla reklamy interstitial – dlaczego mi to nie działa? Wygląda jakby te providery zaskakiwały po chiwli a nie od razu.. + do tego dźwięk na reklamie interstitial jest urywany – - sprawdzenie czy provider od dźwięku jest inicjalizowany w kilku miejscach – powinien być używany tylko i wyłącznie jako provider – do przeanalizowania  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DB KEY: c3ac20eab4173d1b871d8b3a5e4d46513d0d4249201db1a7d1e3f677df7d8a53  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TODO ALL TESTS CHECKLISTA:  
  
- do przetestowania restore purchases po odinstalowaniu aplikacji oraz po instalacji na tym samym koncie google play ale na innym urządzeniu  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- no i finalnie fajnie by było pousuwać te wszystkie pierdoły z konsoli aby ten log był dosyć czysty – to też do przeanalizowania jest, printy na koniec do pokomentowania, do podmiany informacje na temat reklam w odpowiednich miejscach itd.. muszę sobie zapisać co jest do zrobienia kiedy apka już idzie do release + troche refactoringu  
  
- ogolne dokończenie wszystkich formalności w sklepie Google Play itp., zanim aplikacja zostanie wysłana do sprawdzenia + musi zostać sprawdzona jeszcze przez AI googla…

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**KIEDY APKA BĘDZIE JUŻ W SKLEPIE PLAY:**- ekrany reklamujące aplikacje – dla tabletów 7 i 10 cali oraz ich tłumaczenia- wysyłanie powiadomień z poziomu firebase, do zrobienia skrypt pythonowy gdy już będzie troche uzytkownikow aby dobrze tym zarządzać i wysyłać powiadomienia z cloud messaging: wyslac np.: „Hej dość szybko skończyłeś grę ostatnio, napiszesz nam co ci się podoba lub tez nie w naszej aplikacji? Może coś byś poprawił lub chciał nowe funkcje? Napisz nam!”  
//  
ALBO: powiadomienie dla użytkowników kiedy są free, np., czy wiesz, że w pełnej wersji gry rozrywka jest o wiele fajniejsza? Sprawdź! – to będzie do obsłużenia już w apce dlatego że na razie na stałe odpalam stronkę od tamtego powiadomienia

- przemyślenie dlaczego czasem pojawiają się problemy z wyświetlaniem reklam, prawdopodobnie trzeba będzie zarządzać cyklem życia reklamy i tworzyć ją w obrębie kontekstu, który po wyjściu z ekranu mógłby zostać zniszczony – dotyczy to na szczęśćie tylko i wyłącznie małych reklam – natywnych – ogólnie to przy normalnym graniu i testach działa ok – wiec problem dotyczy tylko HOT reaload???

- aplikacja zapisuje dane płatności w SharedPreferences co jest łatwe do zhackowania – ogólnie trzeba będzie przejść na serwer lub dodać dodatkowe mechanizmy weryfikacji – póki co tak zostawiam, skupimy się maksymalnie na wydaniu!

- jak się szybko klika w menu to nadal da się zrobić że wywali out of range – co ciekawe tylko na starszych urzadzniach – raz mi się to udało zrobić na tablecie raz nie – ogólnie tylko wtedy gdy coś zamuli – wstrzymam się z tym chyba az do wersji release bo zapewne wersja zbuildowana nie zmuli?  
  
- Karta improwizacji – TO DO – losowa karta na całą turę, bonusowa z dodatkową animacją „achievmentu” - baza już zrobiona, 70słów per język i po prostu będziemy losować jedno słowo i wyświetlać kartę, dając około minuty czasu na to, możnaby zrobić aby to po prostu pojawiało się absolutnie losowo na całej planszy i tylko raz przez całą turę – czyli, żeby można było dostać tę kartę w każdym momencie gry lecz tylko raz – do przemyślenia jeszcze, ale to by był the best pomysł, dodałoby zaskoczenia grze, losowości i więcej emocji.  
  
- możnaby też dodać info w samouczku oraz do „zasady gry” aby drużyny dobrały się w pary // teoretycznie jest to napisane ale nie tak dokładnie, jeśli będzie jakiś feedback na ten temat to wtedy to zrobie  
  
- ogarnięcie ewentualne koła fortuny tak aby powiększało się trakcie kręcenia na cały ekran, powiększenie planszy tak aby była na całym ekranie – przeorganizowanie czegoś – kwestia do przemyślenia czy w ogóle będzie to możliwe do zrobienia – ewentualnie dla małych ekranów poniżej 5 cali

- żeby pomyslec w przyszłości o android TV, lub webowce aby moc odtwarzac gre na większych ekranach, żeby przyjemniej się gralo (wiec musiałby być tez dodany tryb poziomy aplikacji – must be w tym przypadku! )

- jeżeli aplikacja będzie miała powyżej 0,5mln pobrań wydać ją na iphony + zwiększanie ilości języków, szczególnie na rynek: Japoński, Indie, Korea (tam gdzie jest duży potencjał Google Play ale nie ma blokad cenzuralnych)  
  
- całkowita wymiana planszy ikonek i wszystkiego (szata graficzna) - efekt gry „po północy” – cała logika taka sama ale ze zmianą bazy danych, dodatkowo płatna funkcja aby wymienić grę na tryb „hot” - na funkcji mikropłatności – jak by to miało działać aby był to tryb tylko dla 2 osób ? – logika musiałaby się o wiele bardziej zmienić… ale jest to do zrobienia ogolnie  
  
- możnaby też pomyśleć nad 2 trybem gry, żeby mogły grac tylko 2 osoby, tak jak jest to w gierkach małżeńskich, też możnaby dodać info aby ustaliły kto jest graczem nr.1 kto graczem nr.2 gdyż może to mieć znaczenie w niektórych kartach gdzie aplikacja będzie odnosiła się do danej osoby z drużyny

- odnośnie loading indicator - > możnaby zrobić jakiś customowy indicator, np. kręcące się te strzałeczki z danej karty albo np. kwadracik pokonujący jakiś customowy path, pomyśleć ogólnie czy powinien być w ogole indicator taki stosowany np. pomiędzy przejściami do innych ekranow albo w przypadku każdego pageRoute czy w ogole indicator powinien być pojawiany ?

- pamiętać o iphonach, korygować, odnośnie układu, projektowania etc.. jeszcze można trochę doczytać co jest istotne

**EKRAN SAMOUCZKA: (ewentualność, jeżeli naprawdę nie będzie zrozumienia)**

- zrobienie samouczka przed uruchomieniem gry, informacje dotyczące jego i pomysły:

- dopisanie do planszy scrollingowej (zasady gry) na początku link z alerta do samouczka.. tak aby można było go odbyć w dowolnym momencie

- zamiast po kolei przenosić użytkownika w kodzie, możnaby zrobić jakby „pokaz slajdów” przed uruchomieniem gry -> same widżety i automatyczne przenoszenie, przejścia, klikając.. -> wtedy gdy ktoś by go odbył, dopiero nastąpiłoby uruchomienie głównej planszy.. a gdy wcześniej ktoś odbył to przy odpalaniu gry musiałaby być zawsze sprawdzana flaga załączająca – czyli w sumie ekran ładowania by mógł sterować tym czy wyświetla samouczek czy główny ekran gry

- samouczek musi mieć możliwość ponownego włączenia w ustawieniach, przy pierwszym odpaleniu mu się zmienić odpowiednia flaga, którą w ustawieniach będzie można „cofnąć”

- przekierowanie lub informacja o ponownej możliwością włączenia samouczka, powinna być dostępna z poziomu „zasady gry”

- samouczek ma sens zrobić dopiero wtedy gdy już będę miał gotowy główny ekran całej gry, karty itp. Tak aby po kolei można było pokazywać funkcje, a musi to bazować już na finalnym wyglądzie aplikacji, pamiętaj ze samouczek musi być responsywny!

- wygląd samouczka możnaby zrobić tak, że będzie cały folder z gotowymi klasami, ale bez funkcjonalności (gry), z responsywnością, taki static i wygląd byłby taki, że byłyby narzucone alert dialogie z buttonami – ewentuanymi animacjami oraz z przyciemnieniem tła -> i to tych części których w tym danym momencie nie chciałbym omawiać np. omawiam pola, to pola są kolorowe, reszta jest przyciemniona, na środku móże być krótki opis, a na dole po prawej stronie przycisk w stylu „>>” lub jakiś taki „okrągły”, który mógłby tworzyć efekt ripple czy coś takiego

final Map<FieldType, String> fieldTypes = {  
 FieldType.arrow: 'field\_arrows', // wybór karty  
 FieldType.rhyme: 'field\_sheet', // rymowanie  
 FieldType.alphabet: 'field\_letters', // alfabet, literka  
 FieldType.pantomime: 'field\_pantomime', // pantomimy  
 FieldType.famousPeople: 'field\_microphone', // sławne osoby  
 FieldType.starTaboo: 'field\_taboo', // słowa tabu  
 FieldType.start: 'field\_start', // start  
 FieldType.starBlueDark: 'field\_star\_blue\_dark', // zadanie fizyczne  
 FieldType.starPink: 'field\_star\_pink', // synonimy, antonimy  
 FieldType.starGreen: 'field\_star\_green', // rysowanie  
 FieldType.starYellow: 'field\_star\_yellow', // compare question  
};

**OPIS KART I ICH BAZY DANYCH:**

Baza danych, tabel z kartami, tabela Cards:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Card\_type | Words | Language | IsPurchased | Key+nr np.taboo1 |
| 1 | To samo co w key ale bez numerów |  | PL\_pl | Yes/No | taboo |
| 2 |  |  | DE\_de |  | pantomimes |
| 3 |  |  | EN\_en |  | peoples |
| 4 |  |  | FR\_fr |  | rymes |
| 5 |  |  | IT\_it |  | picture\_taboo |
| 6 |  |  | ES\_es |  | draw\_movie |
| 7 |  |  |  |  | draw\_proverb |
| …. |  |  |  |  | draw\_love\_pos |
|  |  |  |  |  | compare\_question |
|  |  |  |  |  | antonimes |
|  |  |  |  |  | alphabet1 |

Baza danych, tabel z obrazkami picture\_taboo, tabela Images:

|  |  |  |  |
| --- | --- | --- | --- |
| Id | IsPurchased | Image | Key |
| 1 | Zawsze Yes | BLOB - obrazek | picture\_taboo1 |
| 2 |  |  | picture\_taboo2 |
| … |  |  | … |

OPIS KART, karty podstawowe: (czas klepsydry: party time: 52s, gierki małżeńskie: 42s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Karta: | Ilość rekordów per. język | Opis | Ile per runda do czasu: | Mnożnik (5pkt per karta) |
| Taboo | 220 | 1 słowo opisane,5 słów zakazanych | 5 razy na 40 sekund + 2 pominięcia | 1x |
| Pantomimy | 135 | 2 słowa do odgadnięcia per karta | 2 razy na 40 sekund + 1 pominięcie | 2.5x |
| Sławni ludzie | 138 | 5 ludzi | 5 razy 50 sekund + 2 pominięcia | 1x |
| Rymy | 119 | 5 rymów | 5 razy na 50 sekund + 1 pominięcie | 1x |
| Alfabet | Brak w bazie, losowanie z poziomu aplikacji, mam litery w .txt | Animacja losowania liter czy coś, albo po prostu już karta z literą – 20 słów na daną literę | 1 raz bez pominięcia na 50 sekund, brak dolnych przycisków + na koniec musi być pytanie czy się udało czy nie lub opcja pominięcia w trakcie | 0 lub 5 pkt |

OPIS KART, karty płatne:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Karta: | Ilość rekordów | | | | Opis | Ile per runda do czasu: | Mnożnik (5pkt per karta) |
| Taboo obrazkowe, 50/50 karta taboo | 100 obrazków | 100 rekordów słów których nie można użyć |  |  | 1 obrazek,  1 słowo opisane, 5 zakazanych/ 3 warianty per. obrazek | 2 razy per 30 sekund + 1 pominięcie | 2.5x |
| Gwiazdka zieleń - rysowanie | Film 100 | Powiedzenie 88 | Pozycje miłosne 44 |  | Losowanie jednej z kategorii obok | 1 raz per 30 sekund + 1 pominięcie | 0 lub 5 pkt |
| Gwiazdka niebieska zadanie fizyczne | Lista 1: losowanie osoby np. On/Ona  Gracz1/Gracz2 VS animacja? | Lista 2:  Liczby od 1 do 10 | Lista 3: pompki, przysiady, brzuszki, pajacyki + mogą być obrazki na danej karcie (to co wylosuje) | | Najpierw pojawia się pytanie : odpowiada 1 osoba, potem odpowiada 2 osoba i ich compare jeśli równe to +1 | 1 raz per CZAS LOSOWANY z ew. 1 pominięciem, po losowaniu pojawia się co będzie i przycisk „start”, dodatkowy lecący czas | 0 lub 5 pkt |
| Gwiazdka żółta – pytania compare | 250 compare questions | Pytanie;odp1;odp2;odp3 | | |  | Brak czasu, pytanie, możliwość 1 pominięcia tylko na początku | 0 lub 5 pkt |
| Gwiazdka różowa – antonimy/synonimy | 113 slow | Antonimy (1) | Synonimy (3) |  |  | 10s do 3\*8s | 2 + po jednym mnożnik |

+ jeszcze był pomysł na kartę wyzwania z opcją zatwierdzenia czegoś przez innych graczy xD

**POMYSŁ NA PUNKTACJĘ:**

Twoje przemyślenia i analiza dotycząca systemu punktacji są bardzo istotne. Bazując na informacjach, które dostarczyłeś, oto kilka sugestii, które mogą Ci pomóc w zbalansowaniu rozgrywki:

**Modyfikator punktów na podstawie zakręceń:** Twój pomysł z modyfikatorem punktów na podstawie ilości zakręceń może być dobrym rozwiązaniem. Mógłbyś przyjąć formułę, w której podstawowa liczba punktów za zadanie (np. odgadnięcie słów) jest mnożona przez modyfikator oparty na ilości zakręceń, którą drużyna wykonała do tego momentu. To może pomóc wyrównać szanse pomiędzy drużynami, które miały mniej okazji do zdobywania punktów.

**Modyfikator punktów za kartę:** Jeśli karty są zróżnicowane i nie chcesz analizować ich wszystkich, rozważ wprowadzenie modyfikatora punktów, który będzie zależał od liczby słów na karcie. Na przykład, jeśli karta ma mniej słów, jej modyfikator może być wyższy, tak aby całościowa liczba punktów możliwych do zdobycia była podobna dla każdej karty.

**Rundy jako segmenty rozgrywki:** Jeżeli chodzi o podział gry na rundy, możesz to zrobić na wiele sposobów. Jednym z nich może być podział planszy na segmenty, gdzie każdy segment odpowiada rundzie i może wprowadzać nowe zasady lub modyfikatory. Na przykład:

* Pierwszy segment (0-30% planszy): Zadania standardowe, bez modyfikatorów.
* Drugi segment (30-60% planszy): Zadania z dodatkowymi punktami lub specjalnymi wyzwaniami.
* Trzeci segment (60-90% planszy): Zadania z większymi modyfikatorami, wprowadzenie elementów losowych, które mogą zarówno pomóc, jak i utrudnić zdobywanie punktów.
* Ostatni segment (90-100% planszy): Decydujące wyzwania z dużą liczbą punktów lub zasadami, które mogą znacząco zmienić stan gry.

**Wprowadzenie modyfikatorów kart:** Jeśli chcesz, by każda karta miała potencjalnie tyle samo punktów do zdobycia, możesz wprowadzić modyfikator punktów dla każdej karty. Modyfikator ten mógłby zależeć od średniej trudności zadań na karcie, ocenianej na podstawie Twojej wiedzy o grze.

**Cel gry:** Skoro celem gry jest dobra zabawa, warto zastanowić się nad systemem punktacji, który nie przytłacza graczy skomplikowanymi obliczeniami. Uproszczenie systemu punktacji, gdzie skupiasz się na zabawie i interakcji, a nie na ścisłym liczeniu punktów, może być najlepszym rozwiązaniem.

**Testowanie i balansowanie:** Niezależnie od tego, który system wybierzesz, ważne będzie przetestowanie go w praktyce. Możesz zorganizować gry próbne z różnymi grupami graczy i poprosić o feedback. To pomoże Ci zrozumieć, czy system jest sprawiedliwy i czy przyczynia się do zabawy, a także czy jest wystarczająco intuicyjny, aby gracze nie musieli skupiać się na liczeniu punktów, ale mogli się skoncentrować na zabawie.

Mając te informacje, możesz zacząć pracę nad prototypem systemu punktacji, a następnie iteracyjnie go udoskonalać.

DEBUG TOKEN FIREBASE: My debug token  
0CDAE2AE-487E-4B18-ADA9-93479A65D07F

**PLAN:**

### In-App Purchases

1. **Zabezpieczenie dostępu do bazy danych:** Klucz do zaszyfrowania bazy danych powinien być bezpiecznie przechowywany i otrzymywany po autoryzowanym zakupie. Możesz użyć Firebase do przechowywania klucza i sprawdzania, czy zakup został dokonany przez danego użytkownika.
2. **Zapisywanie stanu zakupu:** Użyj SharedPreferences do lokalnego zapisywania stanu zakupu, co pozwoli na zachowanie tej informacji między sesjami aplikacji. Klucz zakupu lub token powinien być również zweryfikowany przez serwer (Firebase), aby uniknąć oszustw.
3. **Restore Purchases:** Musisz zaimplementować funkcję do przywracania zakupów (restore purchases). Wymaga to zwykle integracji z API sklepu aplikacji i obsługi kont użytkowników, których zakupy są zapisane w chmurze (np. Firebase).

### Firebase & Firestore

1. **Przejrzyj main.dart i inne pliki:** Sprawdź, jakie pakiety są już zainstalowane i jak są wykorzystywane. Zapisz sobie notatki na temat tego, co każda biblioteka robi, aby lepiej zrozumieć swój kod i zależności.
2. **Nowe konto Firebase:** Jeśli zaczynasz od nowa, utwórz nowe konto Firebase i skonfiguruj potrzebne usługi takie jak Authentication, Firestore, i inne, które będą używane do zarządzania danymi użytkowników i zakupami.
3. **Przechowywanie ID użytkowników:** Przechowuj ID zakupów i użytkowników w Firestore, aby umożliwić restore purchases. Możesz użyć Firebase Authentication, aby identyfikować użytkowników na różnych urządzeniach i platformach.

### Obsługa błędów i testowanie

1. **Obsługa błędów:** Twój kod powinien obsługiwać różne scenariusze, takie jak błędy sieciowe, anulowanie zakupu i inne błędy płatności. Użyj try-catch i odpowiednich callbacków do obsługi tych sytuacji.
2. **Testowanie płatności:** Użyj środowiska testowego sklepu (np. Google Play Console) do przeprowadzania zakupów testowych. Skonfiguruj testowe konta w konsoli Google Play, aby symulować różne scenariusze zakupów.

### Reklamy AdMob

1. **Integracja AdMob:** Przejrzyj dokumentację AdMob, aby zrozumieć, jak zintegrować SDK reklamowe i wyświetlać reklamy w swojej aplikacji.
2. **Usunięcie reklam po zakupie:** Możesz ukryć reklamy po zakupie przez użytkownika opcji bez reklam. Wykorzystaj SharedPreferences do zapisania stanu zakupu i warunkowo wyświetlaj reklamy w UI.
3. **Konfiguracja Android Manifest i Gradle:** Sprawdź, czy twoje pliki konfiguracyjne są poprawnie ustawione dla AdMob. Musisz dodać odpowiednie metadane i zależności do swoich plików AndroidManifest.xml i build.gradle.

Jeśli masz konkretny kod lub konfigurację, z którą potrzebujesz pomocy, proszę przedstaw szczegóły, a postaram się odpowiedzieć bardziej szczegółowo. Pamiętaj, że rozwój aplikacji to proces iteracyjny, więc ważne jest, aby podzielić swoje zadania na zarządzalne części i testować każdą funkcję osobno, zanim połączysz je razem.

Zbudowanie dobrze funkcjonującej aplikacji z zakupami w aplikacji, integracją z Firebase i AdMob jest dość złożonym zadaniem. Aby ułatwić sobie pracę, zaleca się podzielić proces na kilka etapów i skupić się na nich pojedynczo. Oto sugestia, jak można to zorganizować:

**Krok 1: Podstawowa Funkcjonalność Aplikacji**

Zacznij od podstawowej funkcjonalności aplikacji bez zakupów w aplikacji, Firebase i AdMob. Upewnij się, że wszystkie inne funkcje działają prawidłowo. To tworzy solidną bazę do dalszego rozwoju.

### Krok 2: Integracja z Firebase

### Następnie dodaj Firebase do swojej aplikacji. To pozwoli Ci na zarządzanie danymi użytkowników, autentykację oraz inne potrzebne funkcje:

### Skonfiguruj Authentication, jeśli planujesz śledzić indywidualnych użytkowników.

### Ustaw Firestore lub Realtime Database, jeśli potrzebujesz przechowywać dane w chmurze.

**Krok 3: Zakupy w Aplikacji (In-App Purchases)**

Po skonfigurowaniu Firebase, skup się na implementacji zakupów w aplikacji:

* Zaimplementuj i przetestuj podstawowy przepływ zakupów w aplikacji, w tym obsługę sukcesu, błędów i przywracania zakupów.
* Użyj Firebase do weryfikacji zakupów i przechowywania informacji o transakcjach.

**Krok 4: AdMob i Reklamy**

Gdy już masz działające zakupy w aplikacji i Firebase, możesz dodać AdMob:

* Zintegruj AdMob zgodnie z dokumentacją i dodaj reklamy testowe.
* Zaimplementuj logikę ukrywania reklam dla użytkowników, którzy dokonali zakupów w aplikacji.

**Krok 5: Bezpieczeństwo i Testowanie**

Zajmij się bezpieczeństwem swoich transakcji i danych:

* Weryfikuj zakupy po stronie serwera, aby uniknąć oszustw.
* Przetestuj cały przepływ zakupów, przywracania zakupów i wyświetlania reklam w różnych scenariuszach i warunkach sieciowych.

**Krok 6: Optymalizacja i Refaktoryzacja**

Po zaimplementowaniu wszystkich funkcji i przeprowadzeniu testów, przeanalizuj swój kod:

* Zrefaktoryzuj i optymalizuj kod, aby był czysty i łatwy do utrzymania.
* Upewnij się, że kod jest dobrze udokumentowany i przygotowany na przyszłe aktualizacje.

**Krok 7: Publikacja i Monitorowanie**

Po udanym wdrożeniu wszystkich funkcji i dokładnym przetestowaniu, możesz przystąpić do publikacji:

* Monitoruj zachowanie aplikacji na żywo, zbieraj feedback użytkowników.
* Wykorzystaj analitykę Firebase do monitorowania zachowania użytkowników i wydajności reklam.

Każdy z tych kroków wymaga indywidualnego podejścia i testowania. Ostatecznie, najlepszy sposób na rozpoczęcie zależy od stanu aktualnego rozwoju aplikacji i Twojego doświadczenia z każdym z systemów. Jeśli na przykład masz już doświadczenie z Firebase, może okazać się naturalne zacząć od tego punktu. Z drugiej strony, jeśli Twoja aplikacja jest już funkcjonalna i chcesz tylko dodać monetyzację, możesz zacząć od integracji zakupów w aplikacji lub AdMob.