Dokumentacja Projektu Aplikacji Serwis Samochodowy

Cel dokumentu: Dokumentacja ma na celu przedstawienie szczegółowych informacji na temat projektu aplikacji serwisu samochodowego, włączając w to wymagania, architekturę, szczegóły implementacji oraz metody testowania.

Wymagania Funkcjonalne

|  |  |  |
| --- | --- | --- |
| ID | Funkcjonalność | Opis |
| 1 | Logowanie | Umożliwienie użytkownikowi logowania do systemu. |
| 2 | Wyświetlanie usług | Wyświetlanie listy dostępnych usług serwisowych. |
| 3 | Rezerwacja usług | Możliwość rezerwacji wybranej usługi przez użytkownika. |
| 4 | Wybór daty usługi | Umożliwienie wyboru daty realizacji usługi przez kalendarz. |
| 5 | Powrót do głównego menu | Przekierowanie do głównej aktywności po złożeniu rezerwacji. |

Wymagania Niefunkcjonalne

|  |  |  |
| --- | --- | --- |
| ID | Wymaganie | Opis |
| 1 | Wydajność | Aplikacja powinna odpowiedzieć na każde żądanie użytkownika w ciągu 2 sekund. |
| 2 | Skalowalność | System powinien być w stanie obsłużyć wzrost liczby użytkowników i usług. |
| 3 | Bezpieczeństwo | Dane użytkowników muszą być chronione przed nieautoryzowanym dostępem. |

Implementacja

Warstwa logiki biznesowej

Klasy Serwisowe: Opis logiki biznesowej odpowiedzialnej za przetwarzanie rezerwacji usług, zarządzanie danymi użytkowników itp.

Warstwa prezentacji

MainActivity: Wyświetlanie dostępnych usług.

BookingActivity: Formularz rezerwacji usługi.

Metody testowania

|  |  |  |
| --- | --- | --- |
| Typ testu | Narzędzia | Opis |
| Testy jednostkowe | JUnit | Testowanie indywidualnych komponentów logiki aplikacji. |
| Testy UI | Espresso | Automatyczne testy interfejsu użytkownika. |

Dokumentacja Techniczna Kodu

Ogólny Opis Architektury

Aplikacja składa się z dwóch głównych aktywności: MainActivity i BookingActivity, które odpowiadają za prezentację listy usług i proces rezerwacji usług. Aplikacja wykorzystuje prostą bazę danych SQLite do przechowywania danych o usługach i rezerwacjach.

MainActivity

Cel: Wyświetlanie listy dostępnych usług serwisowych.

Kluczowe Metody:

initializeServiceList(): Metoda inicjalizująca listę usług. Wczytuje dane z bazy danych i aktualizuje adapter RecyclerView.

setupRecyclerView(): Konfiguruje RecyclerView, ustawiając LayoutManager i adapter.

BookingActivity

Cel: Umożliwienie użytkownikowi dokonania rezerwacji wybranej usługi.

Kluczowe Metody:

showDatePickerDialog(): Wyświetla dialog wyboru daty, umożliwiając użytkownikowi wybór preferowanej daty usługi.

confirmBooking(): Przetwarza dane wprowadzone przez użytkownika i dokonuje rezerwacji usługi. Waliduje dane i zapisuje rezerwację w bazie danych.

Model Danych

Service: Reprezentuje pojedynczą usługę serwisową. Zawiera takie pola jak id, name, description i price.

Booking: Reprezentuje rezerwację usługi. Zawiera informacje o wybranej usłudze, dacie rezerwacji oraz dodatkowe uwagi od użytkownika.

Baza Danych

Wykorzystuje SQLite do przechowywania danych o usługach i rezerwacjach.

Schemat Bazy Danych:

Tabela services: przechowuje informacje o usługach.

Tabela bookings: przechowuje informacje o rezerwacjach.

Testowanie

Testy Jednostkowe:

Klasy ServiceTest i BookingTest do testowania logiki biznesowej związanej z usługami i rezerwacjami.

Testy UI:

Wykorzystanie Espresso do testowania interakcji użytkownika z interfejsem aplikacji, w szczególności procesu rezerwacji.

Użyte Biblioteki

RecyclerView dla wyświetlania listy usług.

DatePickerDialog do wyboru daty przez użytkownika.

SQLiteOpenHelper do zarządzania bazą danych.

Przykładowe Fragmenty Kodu

// Przykład metody confirmBooking w BookingActivity

private void confirmBooking() {

if (selectedDate.isEmpty()) {

Toast.makeText(this, "Proszę wybrać datę.", Toast.LENGTH\_SHORT).show();

return;

}

// Logika zapisu rezerwacji do bazy danych

}

Wnioski i Dalsze Kierunki Rozwoju

Aplikacja może być rozwijana o dodatkowe funkcjonalności, takie jak zarządzanie profilami użytkowników, obsługa płatności czy integracja z zewnętrznymi systemami serwisowymi.

Wnioski i Dalsze Kierunki Rozwoju

Aplikacja może być rozwijana o dodatkowe funkcjonalności, takie jak zarządzanie profilami użytkowników, obsługa płatności czy integracja z zewnętrznymi systemami serwisowymi.