Wyniki etapu IV: Implementacja i testy

System monitorowania przejazdów pojazdów autonomicznych „Al-Cab”

Projektowanie oprogramowania

Skład zespołu: Prowadzący:

Aleksander Stepaniuk 272644

Kacper Zakrzewski 272693

Bartosz Dusza 272704

Implementacja

# Struktura kodu

Obraz zawierający tekst, zrzut ekranu, menu, Czcionka

Opis wygenerowany automatycznie

# Generacja kodu

Niektóre części kodu źródłowego zostały wygenerowane automatycznie przez **Flutter SDK** podczas korzystania z poleceń takich jak flutter create oraz flutter pub get.

# Wzorce projektowe

Podczas implementacji aplikacji wykorzystano wzorzec projektowy **Provider** zgodny z architekturą Fluttera. Wzorzec ten umożliwił efektywne zarządzanie stanem aplikacji, ułatwiając komunikację między widżetami bez konieczności przekazywania danych przez wiele poziomów hierarchii komponentów. Dzięki temu aplikacja jest bardziej modułowa, skalowalna i łatwiejsza w utrzymaniu.

Testowanie

# Testy systemowe – przypadki/procedury testowe

## Wybór pojazdu i miejsca docelowego przejazdu (Aleksander Stepaniuk 272644)

|  |  |  |
| --- | --- | --- |
| TC ID: 1  Idea testowa: Początkowy stan MapScreen  Warunki wstępne: Uruchomienie widoku MapScreen | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran MapScreen. 2. Aplikacja wczytuje początkowe dane ekranu. 3. Wyświetlane są domyślne wartości (np. cena 0.00 PLN, długość trasy N/A km, czas podróży N/A min). | Wybrany pojazd | Widoczne elementy: obrazek mapy bez zaznaczonej trasy, zdjęcie wybranego pojazdu oraz jego bardziej szczegółowe dane takie jak rocznik pojazdu, cena za przejazd 0.00 PLN, długość i czas trasy N/A. |

|  |  |  |
| --- | --- | --- |
| TC ID: 2  Idea testowa: Wybór trasy aktualizuje stan  Warunki wstępne: Widok MapScreen jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran MapScreen. 2. Użytkownik wybiera punkt na mapie, klikając w jego obszar. 3. Aplikacja wyświetla zaktualizowaną trasę, cenę, długość, czas oraz obraz mapy. | Wybrany pojazd | Zaktualizowana mapa oraz poprawnie obliczona cena, długość i czas podróży. |

|  |  |  |
| --- | --- | --- |
| TC ID: 3  Idea testowa: Zatwierdzenie bez wybranej trasy wyświetla dialog błędu  Warunki wstępne: Widok MapScreen jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran MapScreen. 2. Użytkownik próbuje zatwierdzić trasę bez wyboru punktów na mapie. 3. Aplikacja wyświetla komunikat o błędzie "Nie wybrano żadnej trasy". | Wybrany pojazd | Wyświetlenie komunikatu o braku wybranej trasy. |

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

## Rejestracja użytkownika (Aleksander Stepaniuk 272644)

|  |  |  |
| --- | --- | --- |
| TC ID: 4  Idea testowa: Walidacja formularza – puste pola  Warunki wstępne: Widok RegisterFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran rejestracji. 2. Użytkownik próbuje zatwierdzić formularz bez wypełnienia żadnych pól. 3. Aplikacja wyświetla komunikaty błędów przy każdym polu. | Brak danych w polach formularza | Komunikaty walidujące dla każdego pola, które jest puste |

|  |  |  |
| --- | --- | --- |
| TC ID: 5  Idea testowa: Rejestracja – adres email już istnieje  Warunki wstępne: Widok RegisterFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran rejestracji. 2. Użytkownik wprowadza dane, w tym istniejący już adres e-mail. 3. Aplikacja wyświetla komunikat o błędzie "Adres email już istnieje w bazie danych". | Dane formularza z istniejącym adresem email | Komunikat: „Adres email już istnieje w bazie danych” |

|  |  |  |
| --- | --- | --- |
| TC ID: 6  Idea testowa: Rejestracja – brak akceptacji regulaminu  Warunki wstępne: Widok RegisterFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran rejestracji. 2. Użytkownik wypełnia wszystkie pola, ale nie zaznacza zgody na regulamin. 3. Aplikacja wyświetla komunikat "Musisz zaakceptować regulamin". | Brak zaznaczonego checkboxa w formularzu | Komunikat: „Musisz zaakceptować regulamin” |

|  |  |  |
| --- | --- | --- |
| TC ID: 7  Idea testowa: Pomyślna rejestracja – wyświetlenie dialogu sukcesu i reset pól  Warunki wstępne: Widok RegisterFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik otwiera ekran rejestracji. 2. Użytkownik poprawnie wypełnia formularz rejestracyjny i zatwierdza go. 3. Aplikacja wyświetla komunikat sukcesu "Konto zostało pomyślnie utworzone!" i resetuje pola formularza. | Poprawne dane formularza | Komunikat: „Konto zostało pomyślnie utworzone!” oraz reset pól |

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

## Płatność za przejazd (Kacper Zakrzewski 272693)

|  |  |  |
| --- | --- | --- |
| TC ID: 8  Idea testowa: Sprawdzenie poprawnej realizacji automatycznej płatności  Warunki wstępne:  - Widok PaymentFragment jest aktywny  - Użytkownik posiada skonfigurowaną automatyczną płatność | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik przechodzi do ekranu podsumowania przejazdu  2. Aplikacja wyświetla powiadomienie o automatycznej płatności  3. Na emaila użytkownika wysłana zostaje faktura  4. Aplikacja wyświetla powiadomienie o potwierdzeniu płatności  5. Aplikacja rozpoczyna przejazd | Płatność automatyczna = TAK | - Ekran informujący o automatycznej płatności  - Ekran potwierdzający płatność  - E-mail z faktura na skrzynce pocztowej użytkownika |

|  |  |  |
| --- | --- | --- |
| TC ID: 9  Idea testowa: Sprawdzenie poprawnej realizacji ręcznej płatności  Warunki wstępne:  - Widok PaymentFragment jest aktywny  - Użytkownik nie ma ustawionej automatycznej płatności | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik przechodzi do ekranu podsumowania przejazdu  2. Użytkownik wybiera metodę płatności  3. Użytkownik wprowadza dane do płatności  4. Aplikacja wyświetla powiadomienie o potwierdzeniu płatności  5. Na emaila użytkownika wysłana zostaje faktura  6. Aplikacja rozpoczyna przejazd | Poprawne dane płatności | - Ekran potwierdzający płatność  - E-mail z faktura na skrzynce pocztowej użytkownika |

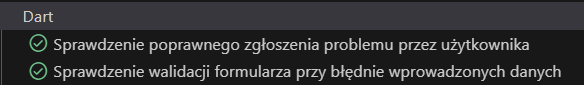
|  |  |  |
| --- | --- | --- |
| TC ID: 10  Idea testowa: Sprawdzenie obsługi błędu płatności  Warunki wstępne:  - Widok PaymentFragment jest aktywny  - Użytkownik wybiera ręczną płatność | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik przechodzi do ekranu podsumowania przejazdu  2. Użytkownik wybiera metodę płatności  3. Użytkownik wprowadza dane do płatności  4. Aplikacja wyświetla powiadomienie o błędzie w płatności | Niepoprawne dane płatności | - Ekran informujący o błędzie w płatności |



## Zgłaszanie problemów i wsparcie użytkownika (Kacper Zakrzewski 272693)

|  |  |  |
| --- | --- | --- |
| TC ID: 11  Idea testowa: Sprawdzenie poprawnego zgłoszenia problemu przez użytkownika  Warunki wstępne:  - Widok SupportFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik przechodzi do sekcji „Wsparcie użytkownika”  2. Użytkownik wypełnia formularz poprawnymi danymi  3. Użytkownik klika „Wyślij”  4. Aplikacja wyświetla komunikat o przyjętym zgłoszeniu i przewidywanym czasie oczekiwania | Poprawne dane formularza | Potwierdzenie przyjęcia zgłoszenia |

|  |  |  |
| --- | --- | --- |
| TC ID: 12  Idea testowa: Sprawdzenie walidacji formularza przy błędnie wprowadzonych danych  Warunki wstępne:  - Widok SupportFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Użytkownik przechodzi do sekcji „Wsparcie użytkownika”  2. Użytkownik wypełnia formularz niepoprawnymi danymi  3. Aplikacja wyświetla komunikat o wprowadzeniu niepoprawnych danych | Niepoprawne dane formularza | Komunikat o braku poprawności danych |



## Monitorowanie stanu technicznego pojazdu (Bartosz Dusza 272704)

|  |  |  |
| --- | --- | --- |
| TC ID: 13  Idea testowa: Sprawdzenie, czy administrator może poprawnie wyświetlić listę wszystkich pojazdów  Warunki wstępne:  - Użytkownik ma uprawnienia administratora floty  - Widok ManageFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Administrator otwiera ekran monitorowania floty  2. System wyświetla listę wszystkich pojazdów | Lista pojazdów we flocie | Wyświetlona lista pojazdów |

|  |  |  |
| --- | --- | --- |
| TC ID: 14  Idea testowa: Sprawdzenie, czy administrator może poprawnie zmienić status pojazdu  Warunki wstępne:  - Użytkownik ma uprawnienia administratora floty  - Widok ManageFragment jest aktywny  - Lista pojazdów jest poprawnie wyświetlona | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Administrator wybiera jeden z pojazdów i przyciska „zmień status”  2. Aplikacja wyświetla szczegóły pojazdu z przyciskiem do zmiany statusu  3. Administrator przy pomocy tego przycisku zmienia status  4. Administrator przyciskiem zatwierdza zmiany we flocie | Dane wybranego pojazdu | Wyświetlone szczegóły pojazdu,  Zmieniony status pojazdu |

## Zakup subskrypcji firmowej (Bartosz Dusza 272704)

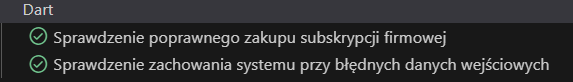
|  |  |  |
| --- | --- | --- |
| TC ID: 15  Idea testowa: Sprawdzenie poprawnego zakupu subskrypcji firmowej  Warunki wstępne:  - Klient firmowy nie ma aktywnej subskrypcji  - Klient posiada wymagane dane do zakupu  - Widok SubFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Klient firmowy otwiera ekran zakupu subskrypcji  2. Klient wprowadza wymagane dane  3. Klient akceptuje regulamin  4. Aplikacja wyświetla dostępne metody płatności  5. Klient wybiera metodę płatności  6. Klient wprowadza poprawne dane płatności  7. Klient finalizuje płatność  8. System wysyła powiadomienie e-mail i SMS | Poprawne dane klienta i płatności | Wyświetlenie komunikatu o pomyślnym zakupie subskrypcji  Powiadomienie na e-mail i SMS |

|  |  |  |
| --- | --- | --- |
| TC ID: 16  Idea testowa: Sprawdzenie zachowania systemu przy błędnych danych wejściowych  Warunki wstępne:  - Klient firmowy wprowadza niepoprawne dane  - Widok SubFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Klient firmowy otwiera ekran zakupu subskrypcji  2. Klient wprowadza wymagane dane  3. Klient akceptuje regulamin  4. Aplikacja wyświetla komunikat o niepoprawnych danych wejściowych | Niepoprawne dane klienta | Wyświetlenie komunikatu o niepoprawnych danych wejściowych |

|  |  |  |
| --- | --- | --- |
| TC ID: 17  Idea testowa: Sprawdzenie zachowania systemu, gdy klient ma już subskrypcję  Warunki wstępne:  - Klient posiada już aktywną subskrypcję  - Widok SubFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Klient firmowy otwiera ekran zakupu subskrypcji  2. Aplikacja wykrywa aktywną subskrypcję  3. Aplikacja przekierowuje użytkownika do ekranu zarządzania subskrypcją | Dane aktywnej subskrypcji firmowej | Przekierowanie do ekranu zarządzania subskrypcją |

|  |  |  |
| --- | --- | --- |
| TC ID: 18  Idea testowa: Sprawdzenie reakcji systemu, gdy użytkownik nie akceptuje regulaminu  Warunki wstępne:  - Klient firmowy odmawia akceptacji regulaminu  - Widok SubFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Klient firmowy otwiera ekran zakupu subskrypcji  2. Klient wprowadza dane, ale nie akceptuje regulaminu  3. Aplikacja wyświetla komunikat: „Aby zakupić subskrypcję, musisz zaakceptować regulamin.”  4. Klient akceptuje regulamin  5. Proces zakupu jest kontynuowany |  | Komunikat o potrzebie zaakceptowania regulaminu,  Kontynuacja procesu zakupu po zaakceptowaniu |

|  |  |  |
| --- | --- | --- |
| TC ID: 19  Idea testowa: Sprawdzenie reakcji systemu, gdy płatność nie powiedzie się  Warunki wstępne:  - Klient firmowy wybiera metodę płatności  - Występuje błąd podczas przetwarzania płatności  - Widok SubFragment jest aktywny | | |
| Krok testu | Dane testowe | Oczekiwane wyniki |
| 1. Klient firmowy otwiera ekran zakupu subskrypcji  2. Klient wprowadza wymagane dane  3. Klient akceptuje regulamin  4. Aplikacja wyświetla dostępne metody płatności  5. Klient wybiera metodę płatności  6. Klient wprowadza niepoprawne dane płatności  7. Klient finalizuje płatność  8. Aplikacja wyświetla komunikat o błędzie w przetwarzaniu płatności | Niepoprawne dane płatności | Wyświetlony komunikat o błędzie w przetwarzaniu płatności |

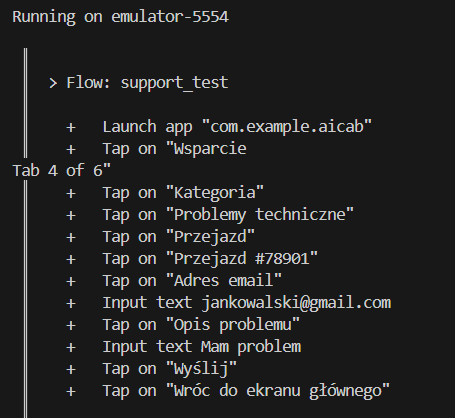


# Testy systemowe automatyczne

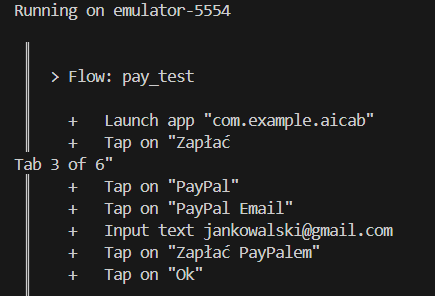
Wszystkie testy zostały zautomatyzowane za pomocą wbudowanego frameworka testowego Fluttera.   
  
Dodatkowo wykorzystaliśmy zewnętrzne oprogramowanie testowe **Maestro** do automatyzacji testów funkcjonalnych:

(W każdym przypadku znak + oznacza pozytywny wynik testu.)

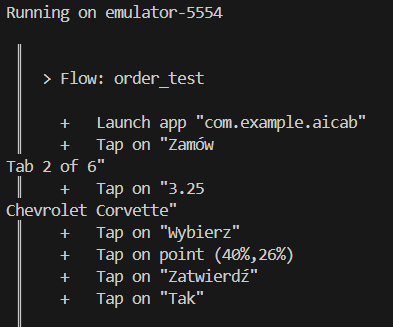
**Support\_test** – test sprawdzający wypełnienie formularza zgłoszenia, a następnie wysłania go.



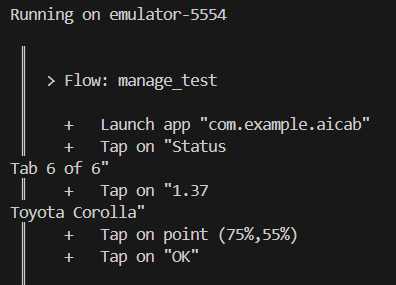
**Pay\_test** – test sprawdzający wybór płatności oraz podanie danych potrzebnych do zatwierdzenia płatności.



**Order\_test** – test sprawdzający wybór samochodu oraz trasy, a następnie zatwierdzenie przejazdu.



**Manage\_test** – test sprawdzający zmianę statusu pojazdu w panelu do zarządzania.



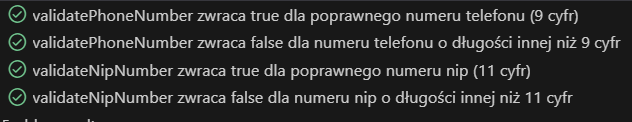
**Register\_test** – test sprawdzający wypełnienie formularza i stworzenie nowego użytkownika.

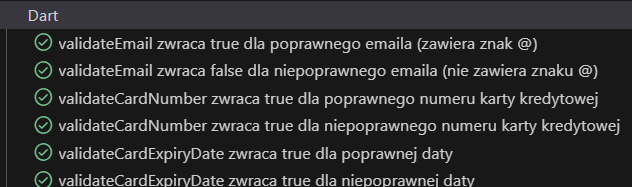


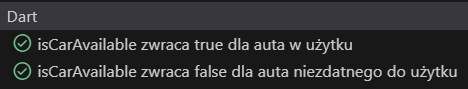
# Testy jednostkowe

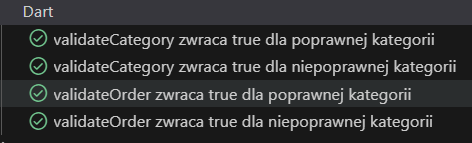
Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie









Łącznie testów: 31 we flutterze + 5 testów w maestro. (wszystkie przechodzą niezależnie od kolejności ich uruchomienia)

