ShotFlow

Końcowa specyfikacja

Michał Kiedrzyński

Styczeń 2025

Spis treści

1 Opis projektu

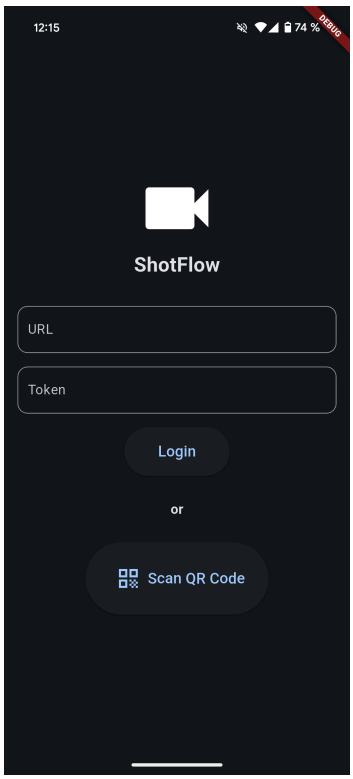
ShotFlow to aplikacja przeznaczona dla zespołów realizacji wizualnej, na wydarzeniach takich jak koncerty, konferencje, czy programy telewizyjne, w których lista kolejnych ujęć może być zaplanowana z wyprzedzeniem. Aplikacja pozwala między innymi na:

- śledzenie postępu realizacji w czasie rzeczywstym,
- dostarczenie każdemu operatorowi informacji o jego ujęciach,
- wyświetlanie informacji czy jest się na wizji niezależnie od aktualnego ekranu,
- szybką zmianę planu realizacji w przypadku zmiany sytuacji,
- bezpośrednią komunikację z realizatorem poprzez czat,
- personalizację wyglądu aplikacji.

1.1 Ekrany

Aplikacja składa się z kilku ekranów, po których nawigacja jest bardzo intuicyjna.

1.1.1 Ekran logowania



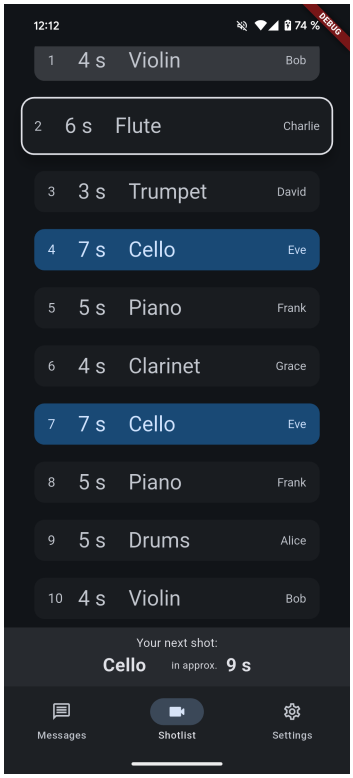
Na ekranie logowania należy wpisać adres WebSocket serwera (zaczynający się od `ws://` lub `wss://`) oraz token autoryzacyjny. Po kliknięciu przycisku *Zaloguj* aplikacja spróbuje połączyć się z serwerem. W przypadku niepowodzenia wyświetli się odpowiedni komunikat.

Dla usprawnienia procesu, możliwe jest również zalogowanie przy pomocy kodu QR, na którym zakodowany jest adres serwera oraz token w formacie JSON:

```
{"url": "ws://example.com","token": "abc"}
```

Kliknięcie w przycisk *Zeskanuj kod QR* otworzy kamerę, która po zeskanowaniu kodu automatycznie wypełni pola adresu serwera i tokenu, a następnie spróbuje połączyć się z serwerem.

1.1.2 Ekran ujęć

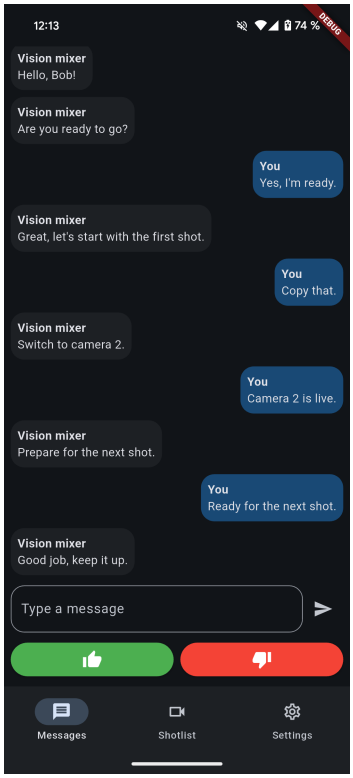


Po zalogowaniu następuje przekierowanie na główny ekran, jakim jest lista ujęć. Wyświetany jest tutaj aktualny stan transmisji, a sama lista przewija się automatycznie w miarę jej postępu. Ujęcia operatora wyróżnione są kolorem (konfigurowalnym w ustawieniach). Na dole ekranu wyświetlany jest pasek z informacją o kolejnym ujęciu i przybliżonym czasie do jego rozpoczęcia.

Użytkownik może również samodzielnie przewinąć listę, aby zobaczyć inne ujęcia. Automatycznie przewijanie jest wtedy wstrzymywane i w prawym dolnym rogu pojawia się przycisk pozwalający na jego wznowienie.

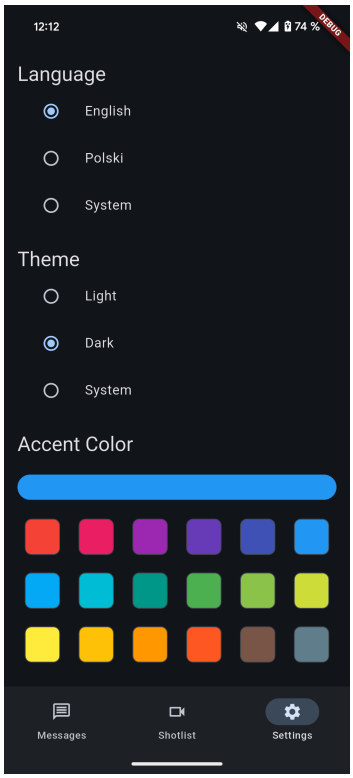
Dodatkowo, niezależnie od aktualnego ekranu, na całą aplikację nakładana jest ramka, która zmienia kolor na zielony, gdy operator jest następny do wejścia, oraz czerwony, gdy jest na wizji.

1.1.3 Ekran wiadomości



Ekran wiadomości pozwala na szybką komunikację z realizatorem w czasie rzeczywistym. Poza standardową metodą wpisywania tekstu, dostępne są również dwa przyciski szybkiej odpowiedzi, aby maksymalnie zaoszczędzić cenny czas w trakcie realizacji.

1.1.4 Ekran ustawień



Ekran ustawień pozwala na zmianę języka aplikacji, dostosowanie jej wyglądu do własnych potrzeb i preferencji, a także wylogowanie (po przewinięciu w dół).

1.2 Komunikacja

System używa własnego serwera, w zamyśle uruchomionego na stacji realizatorskiej. Aplikacja komunikuje się z serwerem przy pomocy protokołu WebSocket. Szyfrowanie komunikacji jest możliwe (protokół WSS), ale nie jest wymagane.

1.2.1 Logowanie

Podczas logowania aplikacja ustanawia połączenie z serwerem, a następnie przesyła token autoryzacyjny. W przypadku niepowodzenia serwer zwraca błąd i zrywa połącznie, a aplikacja wyświetla komunikat o nieudanej próbie logowania. W przypadku powodzenia serwer zwraca informację o zalogowaniu, a aplikacja przechodzi do ekranu ujęć.

1.2.2 Przesłanie listy ujęć

Aby uniknąć problemów z synchronizacją, aplikacja nie przechowuje listy ujęć lokalnie dłużej niż czas jej życia. Dlatego po każdorazowym zalogowaniu serwer wysyła listę ujęć w formacie:

```
{
    "type": "shotlist_update",
    "data": [
        {
        "id": 0,
        "title": "Drums",
        "operator_id": 1,
        "operator_name": "Alice",
        "duration": 5
        },
        {
        "id": 1,
        "title": "Violin",
        "operator_id": 2,
        "operator_name": "Bob",
        "duration": 4
        },
        [...]
    ]
}
```

Taka wiadomość może zostać wysłana również w trakcie realizacji, jeśli lista ujęć zostanie zmieniona.

1.2.3 Przypisanie operatora

Po wysłaniu listy ujęć, serwer nadaje użytkownikom ich identyfikatory, używane do przypisania operatorów do konkretnych ujęć. Dzieje się to poprzez wysłanie wiadomości:

```
{"type": "operator_assign", "operator_id": 5}
```

Tak jak poprzednio, możliwa jest zmiana id operatora w trakcie realizacji.

1.2.4 Przesłanie historii wiadomości

Ze względu na styl komunikacji podczas realizacji (krótkie, szybko przedawniajace się wiadomości) oraz pracę z różnymi realizatorami podczas różnych wydarzeń, aplikacja nie przechowuje historii wiadomości lokalnie. Po zalogowaniu serwer ma jednak możliwość przesłania ostatnich wiadomości, aby zapewnić spójność komunikacji w obrębie jednego wydarzenia. Robi to wysyłając wiadomość:

```
{
    "type": "message_history",
    "messages": [
        {"sender": "Vision mixer", "message": "Hello, Bob!"},
        {"sender": "Vision mixer", "message": "Are you ready to go?"},
        {"sender": "You", "message": "Yes, I'm ready."},
        [...]
    ]
}
```

1.2.5 Zmiana ujęcia

W momencie rozpoczęcia kolejnego ujęcia serwer wysyła wiadomość:

```
{"type": "shotlist_jump", "currently_live": 5}
```

gdzie `currently_live` to id ujęcia, które właśnie się rozpoczyna.

2 Zaimplementowane dodatkowe wymagania

- Aplikacja działa na systemach Android oraz Windows, a także w wersji webowej.
- Ikonki na głównym pasku nawigacyjnym mają customowe animacje.
- Aplikacja posiada testy jednostkowe.
- Logowanie następuje przy pomocy własnej metody autoryzacji z serwerem.
- Na platformach Android oraz Web dostępna jest możliwość logowania przy pomocy kodu QR przy użyciu kamery.
- Aplikacja działa w językach polskim i angielskim.
- Aplikacja przechowuje lokalnie ustawienia użytkownika (język, kolorystyka) przy użyciu *shared_preferences* oraz url serwera i token autoryzacyjny w bezpieczny sposób, przy użyciu *flutter_secure_storage*.

3 Przykładowy serwer

Do aplikacji dołączony jest przykładowy serwer w C#. Serwer ten jest bardzo prosty i służy jedynie do demonstracji działania aplikacji. Wykonuje on sekwencję logowania, a następnie co 3 sekundy przewija listę ujęć. Na wiadomości wysłane przez czat odpowiada echem.

Dla poprawnego działania serwera może być konieczne manualne otworzenie portu w zaporze sieciowej systemu.