U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Programação de Dispositivos Móveis

Projeto

Relatório Técnico

FCUP 2021/2022

Grupo E

Filipe Colla David – 201810097

Introdução

Este projeto teve como objetivo a implementação de uma aplicação para android "TwitchFlix". Esta aplicação consiste num serviço que tenta fundir a Twitch.tv com a Netflix, permitindo ao utilizador ver videos a pedido (*Video on demand*), fazer *streaming* de conteúdo próprio bem como ver *streams* de outros utilizadores. Este projeto está dividido em duas componentes, *frontend* e *backend*. O *frontend* foi desenvolvido usando o IDE da Google, android studio. O backend consiste em 3 componentes, um servidor aplicacional *Jetty+Jersey*, uma base de dados *PostgreSQL* e um servidor nginx. Todos estes sistemas foram implementados localmente, à exceção do armazenamento dos filmes que foi feito na google cloud.

Descrição do sistema

Base de Dados:

A base de dados implementada em *PostgreSQL* consiste em 3 tabelas:

- **Users:** onde são armazenados os dados dos utilizadores, nome de utilizador e password.
- **Movies:** onde são armazenados os dados dos filmes, o título, o ano de lançamento, duração, um link para o armazenamento online do conteúdo e um link para uma *thumbnail* do filme.
- **Streams**: onde são armazenados os dados das *streams* atualmente ativas, que consiste no nome do utilizador e o título da stream.

NGINX:

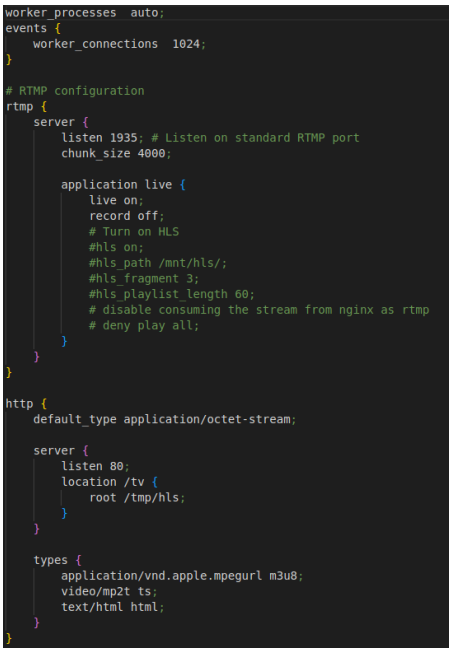
O servidor NGINX foi usado para receber uma *live-stream* e propagar essa mesma *live-stream* com um *delay* mínimo, para tal, foi usado o protocolo RTMP que permite o *streaming* de video e de áudio. Não foi possível concretizar a implementação desta funcionalidade devido a problemas de configuração. No entanto, a configuração usa como *url* rtmp://127.0.0.1:1935:/live/[nome da *stream*], onde 127.0.0.1 seria o *localhost*, 1935 seria a porta de escuta do servidor NGINX, live o nome do servidor/aplicação e o nome da *stream* seria definido pelo *frontend*.

Servidor *Jetty+Jersey*:

O servidor *Jetty+Jersey* gere os diferentes pedidos da aplicação, que podem ser categorizados da seguinte forma:

- **Utilizadores**: gere o registo e login dos utilizadores. O registo é feito usando um nome de utilizador e uma palavra-passe. O nome de utilizador é único na base de dados, funcionalidade que é verificada aquando do registo e caso seja efetuado um registo com um nome de utilizador já existente o mesmo não é permitido. Após o registo com sucesso o nome de utilizador e a password, encriptada usando a biblioteca *security* do java, estes dados são armazenados na base de dados. Para o login os mesmos tipos de dados são inseridos e é feita uma *query* à base de dados para verificar se os dados inseridos coincidem com os dados armazenados.

- **Videos a pedido**: consiste num único *endpoint* onde é feita uma *query* à base de dados e são selecionados todos os filmes para serem posteriormente mostrados ao utilizador na aplicação.
- **Streams**: gere o registo e a propagação das *live-streams* que se encontram ativas. Para tal, tem 3 *endpoints*, um para a criação da stream, um para a remoção da stream, quando esta termina e outro para agregar todas as *live-streams* ativas para serem posteriormente mostrados ao utilizador na aplicação. Para tal, a aplicação envia o nome de utilizador e o nome da string que servem para construir o *url stream*, que é composto pelo link base rtmp://127.0.0.1:1935/live/[nome de utilizador]_[titulo da stream].



```
worker_processes  auto;
events {
    worker_connections  1024;
}

# RTMP configuration
rtmp {
    server {
        listen 1935; # Listen on standard RTMP port
        chunk_size 4000;

        application live {
            live on;
            record off;
            # Turn on HLS
            #hls on;
            #hls_path /mnt/hls/;
            #hls_fragment 3;
            #hls_playlist_length 60;
            # disable consuming the stream from nginx as rtmp
            # deny play all;
        }
    }
}

http {
    default_type application/octet-stream;

    server {
        listen 80;
        location /tv {
            root /tmp/hls;
        }
    }

    types {
        application/vnd.apple.mpegurl m3u8;
        video/mp2t ts;
        text/html html;
    }
}
```

Fig.1 - Configuração do servidor NGINX

Frontend:

O *frontend* para esta app foi dividido em diferentes atividades, de modo a separar as diferentes componentes.

- **Login/Registo:** Para a utilização da aplicação é necessário fazer registo, após o registo o utilizador pode fazer o login com os mesmos dados inseridos para o registo. Caso seja feito um pedido invalido, submeter os campos em branco, nome de utilizador e password errados para o caso de login ou no caso de registo de um nome de utilizador já existente, aparece um Toast a notificar o utilizador destes pedidos inválidos.

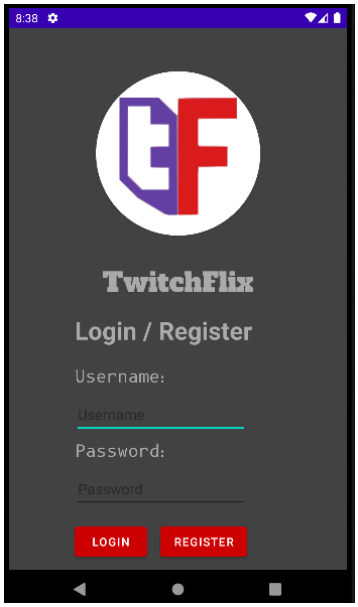


Fig.2 - Login ou Registo do utilizador

- **Menu Principal:** Após o login com sucesso o utilizador é redirecionado para esta atividade onde pode escolher se quer ver videos/filmes (*Watch Videos*), ver *streams* (*Watch* Streams) ou se quer ele próprio fazer *stream* da sua câmera (*Start Streaming*).

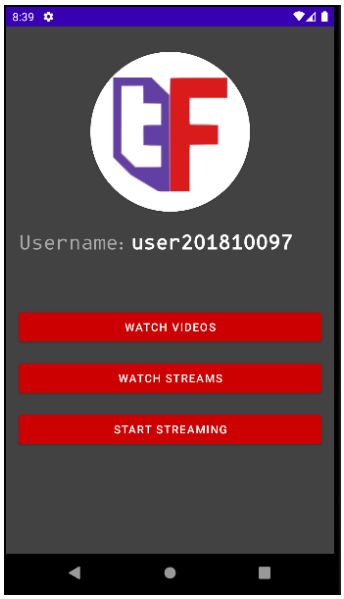


Fig.3 - Menu para a seleção da próxima atividade

- **Videos a pedido (*WATCH VIDEOS*):** Quando o utilizador escolhe esta atividade é direcionado para uma atividade com um catálogo de filmes e quando escolhe um filme é redirecionado para outra atividade onde pode ver o filme, foi ainda implementado um controlador de media para pausar, avançar ou retroceder para diferentes partes do vídeo média. Nesta segunda atividade ficou por implementar a funcionalidade de *fullscreen* e mostrar os diferentes géneros de cada filme (está como *Action* para todos).

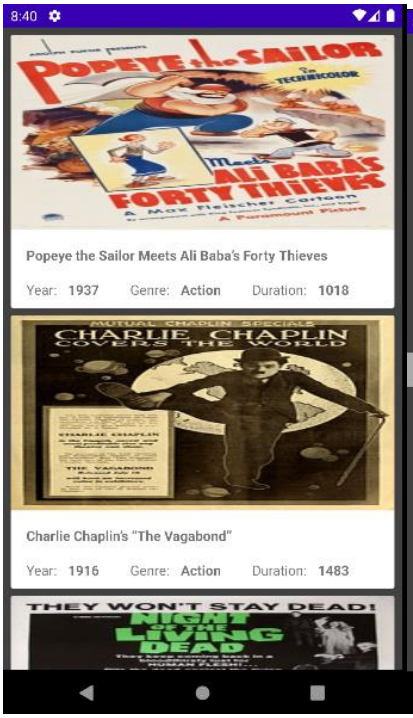


Fig.4 - Catálogo de filmes

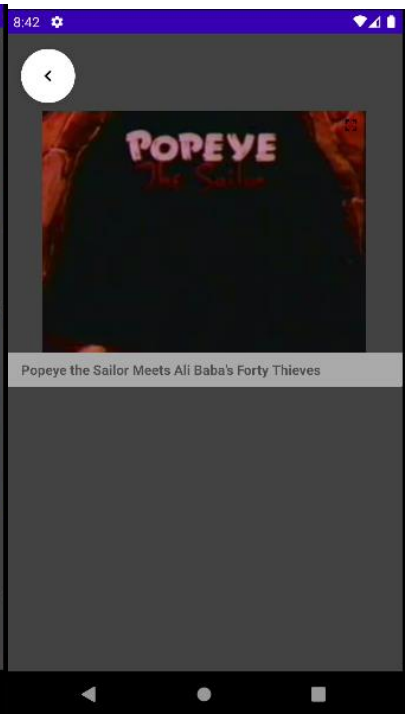


Fig.5 - VideoPlayer

- **Ver Streams (*Watch Streams*):** Nesta componente, o utilizador pode ver as diferentes *streams* que se encontram ativas no servidor *Jetty+Jersey*. No entanto como o servidor NGINX não se encontra funcional estas têm que ser adicionadas fazendo um pedido ao servidor *Jetty+Jersey* que por sua vez adiciona a *stream* à base de dados.

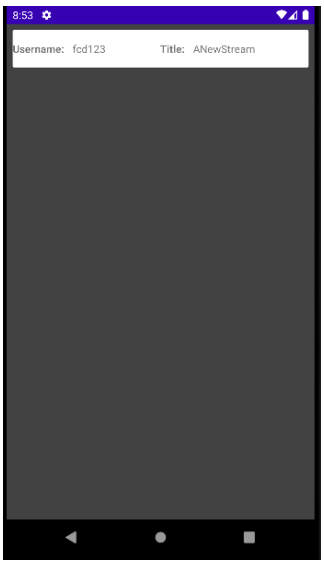


Fig.6 - Catálogo de *Streams*

- **Fazer uma *stream* (*Start Streaming*):** Nesta atividade o utilizador pode inserir o título pretendido para a sua *stream* e quando carrega em *Start Streaming* é redirecionado para uma outra atividade onde procederia ao início da stream carregando no botão e depois à sua eventual paragem, carregando no botão *Start Streaming*, que seria alterado para *Stop Streaming* que serviria para parar a *stream.* Quando o utilizador carregasse em *Start Streaming,* seria feito um pedido POST ao servidor *Jetty+Jersey* para guardar na base de dados esta *stream*, quando fosse carregado o botão *Stop Streaming,* seria feito outro pedido POST ao servidor *Jetty+Jersey* para remover a stream da base de dados. Após armazenada a *stream* na base de dados, esta estaria disponivel para os restantes utilizadores como explicado no componente **Ver Streams (*Watch Streams*).** *No* entanto, não foi possivel implementar esta funcionalidade.

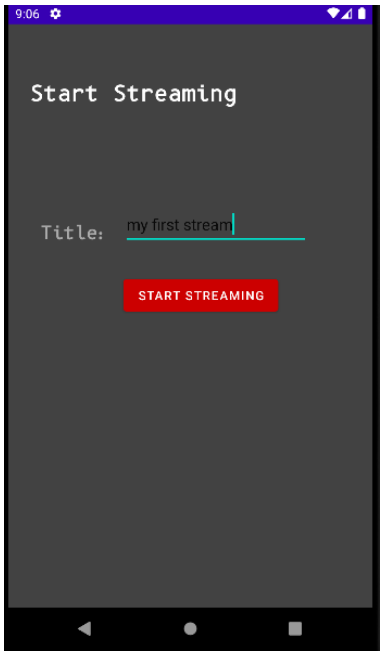


Fig.6 - Escolha do título para a *stream*

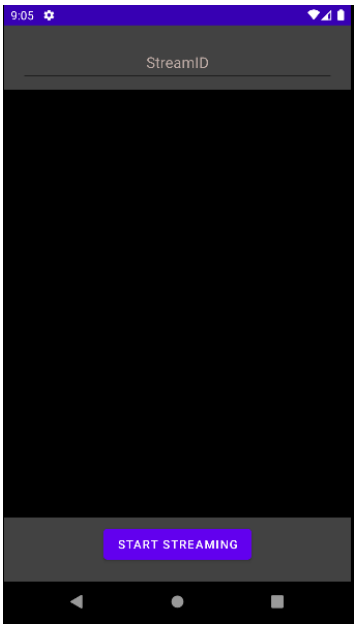


Fig.7 - Stream

Bibliotecas Usadas

Picasso: Uma biblioteca para download de imagens

https://square.github.io/picasso

Rtmp-rtsp-stream-client-java: Biblioteca Android para stream de video e audio para um servidor que usa protocolos RTMP

https://github.com/pedroSG94/rtmp-rtsp-stream-client-java/wiki

OkHttp: Um cliente HTTP e HTTP/2 para Android e aplicações Java

https://square.github.io/okhttp/

Conclusões

Com este projeto tive a minha primeira interação com desenvolvimento de aplicações móveis, não foram atingidos todos os objetivos base, tendo falhado a configuração do servidor NGINX e a implementação do cliente para receber as *streams*. Para próximas versões o objetivo seria implementar as partes em falta bem como melhorar o login e a segurança da aplicação dando a possibilidade ao utilizador de entrar com a conta google, de modo a obter um registo/login mais seguro.