

DESENVOLVIMENTO DE SISTEMAS

EMANUEL OLIVEIRA SALVADOR SOUZA

JOSÉ ALMIRO LIMA DOS SANTOS

LUÍS FELIPE DA SILVA PALMEIRA

**YOUFLIX (DOCUMENTAÇÃO DO APLICATIVO MOBILE)**

Orientador: Rafael Levi

Lauro de Freitas, BA

2020

Sumário

[1. Conceito 3](#_Toc42430002)

[2. Objetivo 3](#_Toc42430003)

[3. Justificativa 3](#_Toc42430004)

[4. Pré-requisitos 3](#_Toc42430005)

[5. Tecnologias 3](#_Toc42430006)

[5.1 Frameworks 3](#_Toc42430007)

[5.2 APIs 4](#_Toc42430008)

[6. Arquitetura do Sistema 4](#_Toc42430009)

[7. Fluxo de funcionamento da aplicação 5](#_Toc42430010)

[8. Armazenamento de dados 6](#_Toc42430011)

[9. Disciplinas Cadastradas 7](#_Toc42430012)

[10. Logo 7](#_Toc42430013)

[11. Conclusão 7](#_Toc42430014)

[12. Código Fonte 8](#_Toc42430015)

[13. Anexos 10](#_Toc42430016)

[14. Referências 11](#_Toc42430017)

# Conceito

O projeto YouFlix trata-se de um aplicativo mobile de streaming de vídeo inspirado no Netflix.

# Objetivo

Desenvolver um aplicativo mobile para organizar videoaulas focado em áreas relacionadas ao curso técnico de Desenvolvimento de Sistemas. Neste conterá as seguintes disciplinas selecionadas:

# Justificativa

Essa aplicação faz parte de uma atividade das aulas de Desenvolvimento Mobile do curso de Desenvolvimento de Sistemas do Senai.

# Pré-requisitos

* Node JS
* Visual Studio Code

# Tecnologias

* React Native
* NPM
* CSS

## Frameworks

* Expo CLI
* Axios

## APIs

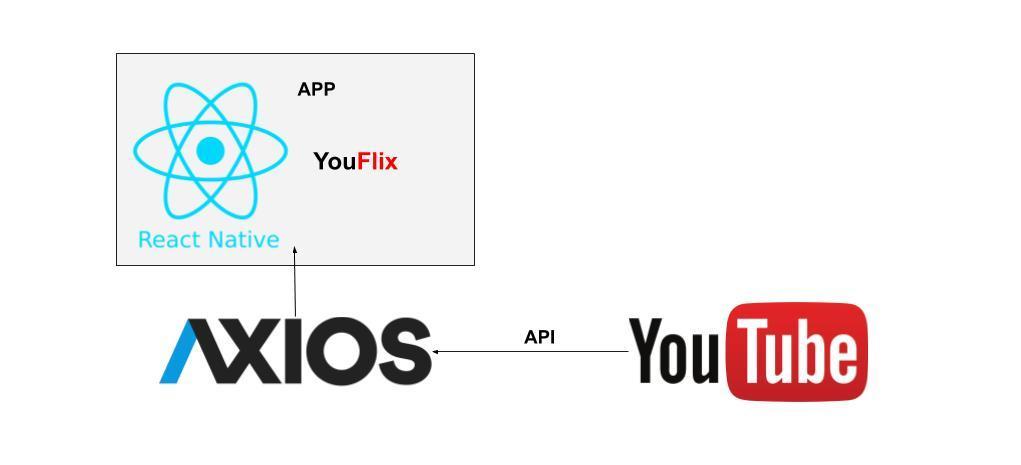
* YouTube Data API

# Arquitetura do Sistema

Foi utilizado o Node JS apenas para ter acesso ao gerenciador de pacotes (NPM). Através do NPM podemos instalar/desinstalar e atualizar dependências do projeto, executar o projeto, e principalmente instalar o Expo CLI. Após a instalação do Expo CLI de forma global, o mesmo pode ser inicializado baixando automaticamente e instalando toda a base do React Native no projeto, inclusive o package.json, que trata-se de um arquivo em formato JSON (JavaScript Object Notation) utilizado para guardar as dependências e *scripts* do projeto. A partir desse ponto só é necessário utilizar/ consumir as APIs necessárias do projeto.

No caso deste projeto (YouFlix) foi utilizado a API do YouTube, disponibilizada pela Google. Para usufruir dessa API foi utilizado o Axios que basicamente trata-se de um cliente HTTP que realiza requisições do tipo *GET* (nesse projeto) para consumir os dados disponibilizados. O CSS é um mecanismo utilizado na estilização do projeto, ou seja, seu objetivo é melhorar o aspecto visual do APP.

Figura 01: Layout da arquitetura do sistema YouFlix



Fonte: Autoria própria.

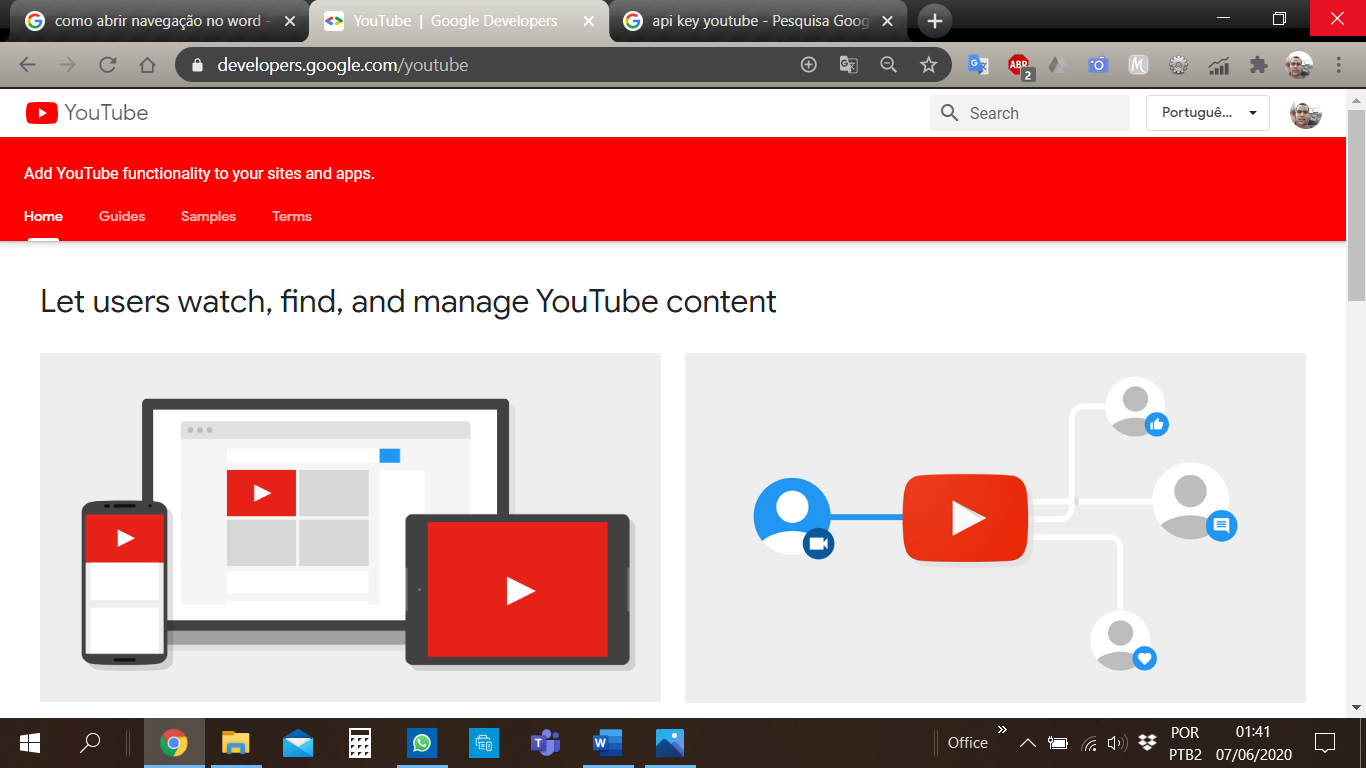
# Fluxo de funcionamento da aplicação

A modelagem básica do aplicativo foi feita a partir da tela home, composta por um arquivo index.js, local onde é localizado o código fonte da página. Posterior a isso, o arquivo style.js fica localizado os arquivos de layout e estilização, utilizando como base o CSS. A página inicial (Home) tem a função de mostrar as seis disciplinas disponíveis, já cadastradas no arquivo Subjects.json.

Na tela home o usuário seleciona uma dessas matérias e automaticamente é redirecionado para página Subjects, que é composta por arquivos index.js e style.js, que apesar de possuírem o mesmo nome e funções, possuem codificação diferente dos seus semelhantes da página Home. A página Subjects apresenta ao usuário as principais informações relaciona aos vídeos da matéria selecionada, tais como: título, descrição e logo (*thumbnails*). Essas informações são obtidas graças a API disponibilizada pelo Google para o YouTube (YouTube Data API), sendo que para fazer essa requisição, é necessário usar o id do vídeo e a chave de API, chamada de API Key.

Após o vídeo ser selecionado, o usuário é redirecionado para a tela player, que também é composta pelos os arquivos index.js e style.js, os quais são encarregados de gerar o player do YouTube. Para o carregamento do player é utilizado a tag html “Embed” que exibe a mídia de forma embarcada.

Figura 02: YouTube Data API



Fonte: <https://developers.google.com/youtube>

Figura 03: Front-End YouFlix



Fonte: Autoria Própria

# Armazenamento de dados

Para fazer as requisições de quais matérias e vídeos o usuário pode assistir, é consultado a pasta Date

O Date possui diversos arquivos json, um arquivo principal chamado Subject, onde todas as disciplinas que vão ser exibidas são cadastras e apontam seu número de id. Os outros arquivos da pasta Date, possuem o id do vídeo especificamente no YouTube, portando, caso um programador queira adicionar outros conteúdos, basta simplesmente manipular esses arquivos json, que funciona de forma menos elaborada como o banco de dados da aplicativo.

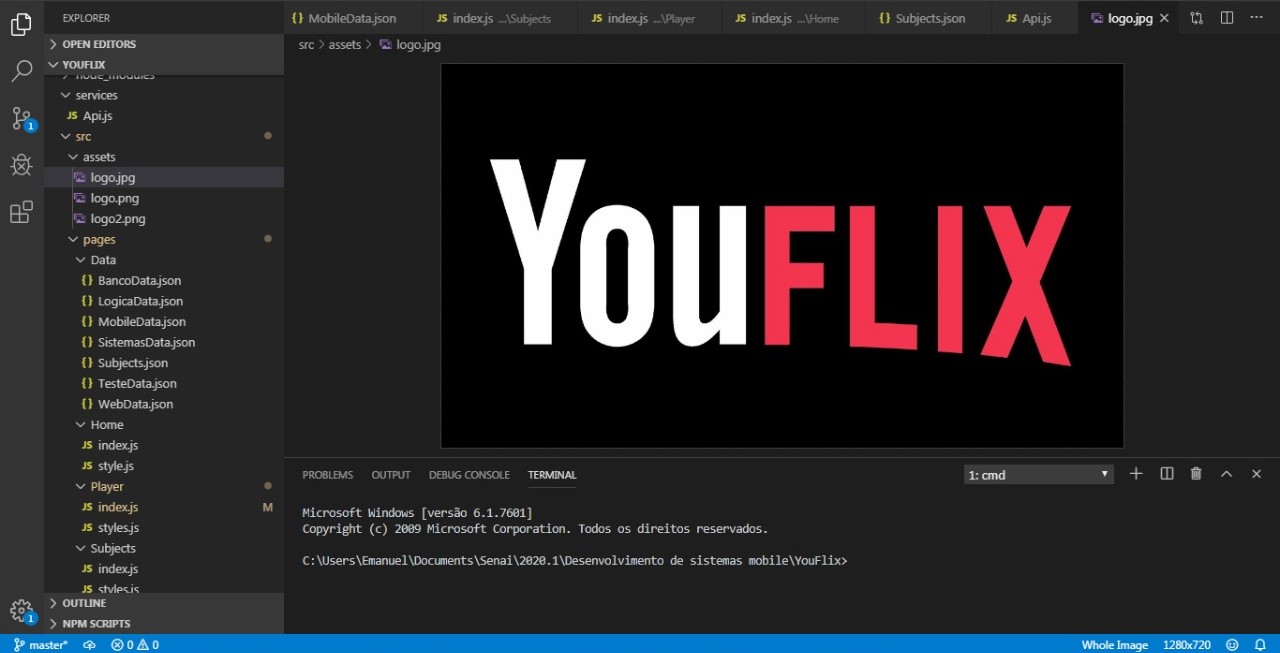
# Disciplinas Cadastradas

* Lógica de programação
* Desenvolvimento de sistemas
* Desenvolvimento mobile
* Banco de dados
* Programação web

# Logo

Dentro do arquivo Assets, consta as logos utilizadas na plataforma. Sendo a principal, a própria logo do YouFlix.

Figura 04: Logo YouFlix



Fonte: Autoria própria.

# Conclusão

. Uma das vantagens de utilizar o React Native e o Expo é a portabilidade, ou seja, a possibilidade de fazer uma única programação (usando JavaScript) para multiplataformas como por exemplo: Android e IOS. Era recomendado utilizar banco de dados relacional (SQL) ou não-relacional (NOSQL), mas devido certas limitações, foi decidido usar os dados dentro do próprio aplicativo.

# Código Fonte

Segue abaixo alguns exemplos de arquivos JSON utilizados no projeto. Para maiores detalhes é necessário verificar o projeto no GitHub (links nas referências).

* Disciplinas:

[

{

"id":"1",

"nameSubjects":"Logica de programação"

},

{

"id":"2",

"nameSubjects":"Desenvolvimento de sistemas"

},

{

"id":"3",

"nameSubjects":"Desenvolvimento mobile"

},

{

"id":"4",

"nameSubjects":"Banco de dados"

},

{

"id":"5",

"nameSubjects":"Programação WEB"

},

{

"id":"6",

"nomeMateria":"Testes de sistemas"

}

]

* Id do Subjects:

[

{

"id":"0",

"videoID":"Ds1n6aHchRU"

},

{

"id":"1",

"videoID":"JLlTo3SwxJE"

},

{

"id":"2",

"videoID":"7ph98Ih\_ckc"

},

{

"id":"3",

"videoID":"vp4jgXA\_BB0"

},

{

"id":"4",

"videoID":"g0iIVeeQo1M"

}

]

* Package

{

"main": "node\_modules/expo/AppEntry.js",

"scripts": {

"start": "expo start",

"android": "expo start --android",

"ios": "expo start --ios",

"web": "expo start --web",

"eject": "expo eject"

},

"dependencies": {

"@react-navigation/native": "^5.5.0",

"@react-navigation/stack": "^5.4.1",

"axios": "^0.19.2",

"expo": "~37.0.3",

"react": "~16.9.0",

"react-dom": "~16.9.0",

"react-native": "https://github.com/expo/react-native/archive/sdk-37.0.1.tar.gz",

"react-native-gesture-handler": "^1.6.1",

"react-native-safe-area-context": "^3.0.2",

"react-native-screens": "^2.8.0",

"react-native-web": "~0.11.7",

"react-native-webview": "^10.2.3"

},

"devDependencies": {

"babel-preset-expo": "~8.1.0",

"@babel/core": "^7.8.6"

},

"private": true

}

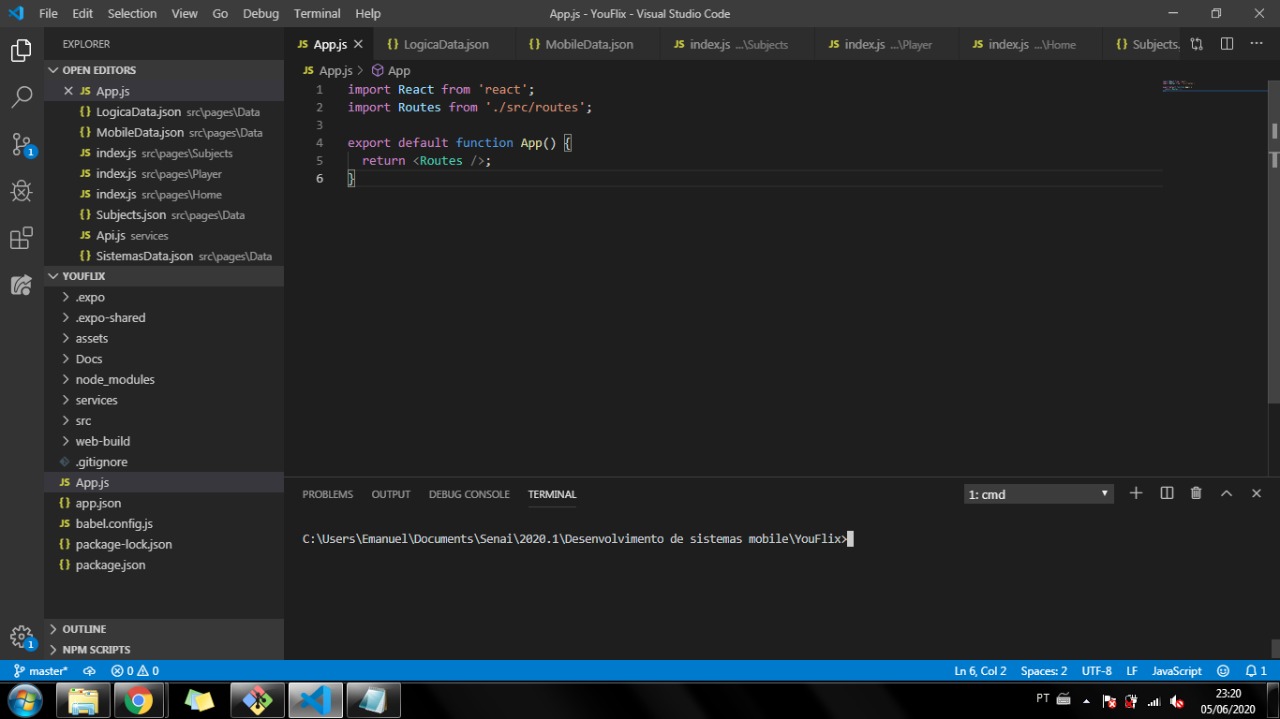
# Anexos

Figura 05: Fotos das páginas do aplicativo



Fonte: Autoria própria.

Figura 06: Página App.js do Visual Studio Code



Fonte: Autoria própria.

# Referências