

MANUAL TECNICO

DISEÑO DESARROLLO E IMPLEMENTACION DE APLICACIÓN
PARA MOVILES YOUAPP

VICTOR ALEJANDRO ALEJO GALVEZ

DENYS ENRIQUE CRUZ INESTROZA

MATTHEW EMILIO GAITAN RAMOS

MARCO ANTONIO HERNANDEZ HERNANDEZ

ALVARO ANIBAL VELASQUEZ RIVAS

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERIA

INGENIERIA DE SISTEMAS

2021

Indice

Presentación	4
Objetivo	5
Procesos	6
Procesos de entrada.....	6
Procesos de salida	6
Herramientas utilizadas para el desarrollo	7
Android Studio.....	7
IntelliJ IDEA.....	7
JDK	7
MySQL	7
NPM.....	7
React native.....	7
Visual Studio Code	7
Docker.....	8
MinIO.....	8
Estructura	9
Estructura del proyecto backend.....	9
Módulo de componentes	10
Docker-images	11
Service-commons.....	11
Services-support.....	11
Services-instances	12
AuthenticationApplication.java	13
StorageApplication.java.....	13
Estructura de proyecto frontend.....	14
src.....	15
Enums	16
Environment.....	16
Models.....	16
Modules.....	17
Screens	17
Utils.....	18
Assets	18
Styles.....	19

Index.js.....	19
---------------	----

Presentación

El siguiente manual guiara a los usuarios que harán soporte a la aplicación YouApp, el cual les dará a conocer los requerimientos y la estructura para la construcción de dicha aplicación, en el desarrollo de la aplicación móvil conectada mediante un backend desarrollado en Java, a una base de datos, el cual muestra las herramientas necesarias para la construcción y la funcionalidad del sistema.

Objetivo

Informar y especificar al usuario la estructura y conformación del sistema con el fin de que puedan hacer soporte y modificaciones o actualizaciones al sistema en general.

Procesos

Procesos de entrada

Ingresar a la aplicación

Ingresar datos de logeo

Ingresar datos para registro de usuarios

Ingresar datos para recuperación de contraseña

Filtrar canciones

Buscar canciones

Crear playlist

Indicar canciones favoritas

Dar me gusta y no me gusta a las canciones

Reproducir de música

Subir canciones

Procesos de salida

Correo para verificación de cuenta

Correo para recuperación de contraseña

Mostrar pantalla home

Lista de canciones filtradas

Resultados de la búsqueda

Mostrar favoritas

Mostrar playlist

Reproducción de música

Datos de las canciones subidas

Herramientas utilizadas para el desarrollo

Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android y está basado en IntelliJ IDEA.

IntelliJ IDEA

IntelliJ IDEA es un entorno de desarrollo integrado (IDE) para el desarrollo de programas informáticos. Es desarrollado por JetBrains (anteriormente conocido como IntelliJ), y está disponible en dos ediciones: edición para la comunidad y edición comercial.

JDK

Java Development Kit (JDK) es un software para los desarrolladores de Java. Incluye el intérprete Java, clases Java y herramientas de desarrollo Java (JDT): compilador, depurador, desensamblador, visor de applets, generador de archivos de apéndice y generador de documentación.

MySQL

MySQL es el sistema de gestión de bases de datos relacional más extendido en la actualidad al estar basada en código abierto.

NPM

De sus siglas **NPM (Node Package Manager)** es un gestor de paquetes desarrollado en su totalidad bajo el lenguaje JavaScript por Isaac Schlueter, a través del cual podemos obtener cualquier librería con tan solo una sencilla línea de código.

React native

React Native es un **framework JavaScript** para crear aplicaciones reales nativas para iOS y Android, basado en la librería de JavaScript React para la creación de componentes visuales, cambiando el propósito de los mismos para, en lugar de ser ejecutados en navegador, **correr directamente sobre las plataformas móviles nativas**, en este caso iOS y Android.

Visual Studio Code

Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

Docker

TI, es una tecnología de creación de contenedores que permite la creación y el uso de contenedores de Linux

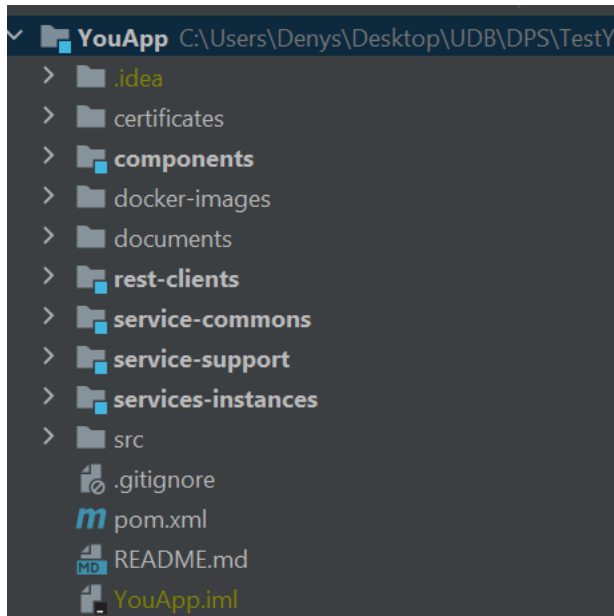
MinIO

Es un servidor de almacenamiento en la nube compatible con Amazon S3, liberado bajo Licencia Apache v2

Estructura

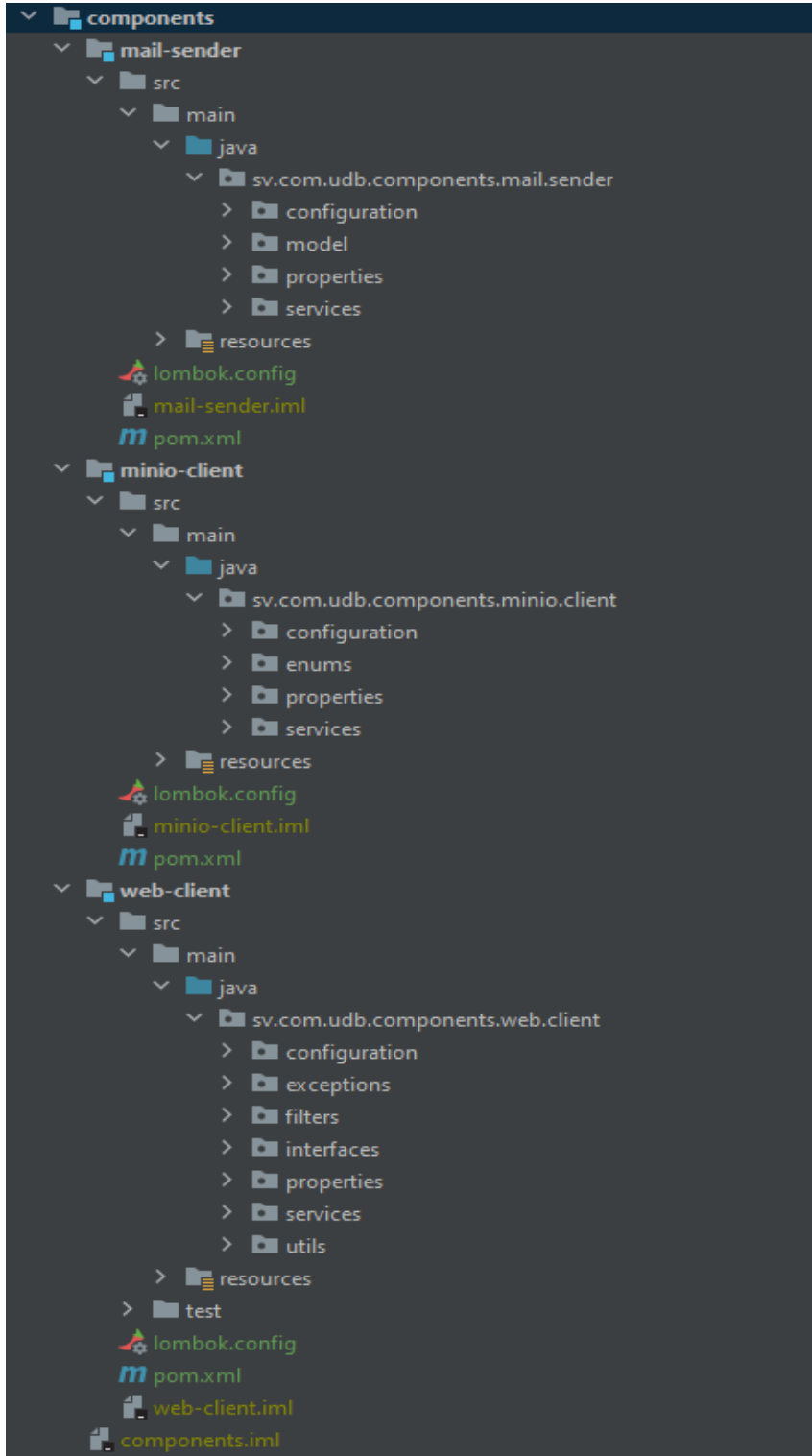
Estructura del proyecto backend

A simple vista podemos observar los módulos padres y carpetas contenedoras de archivos generales del proyecto, entre los módulos tenemos: components, rest-clients, service-commons, service-support, services-instances; cabe mencionar que la mayoría de módulos contiene algunos submódulos para lograr una buena implementación de la arquitectura de microservicios.



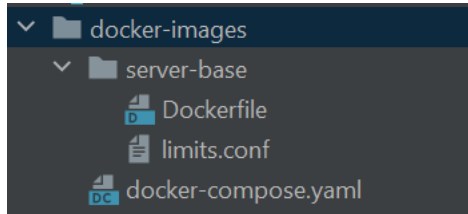
Módulo de componentes

Este módulo contiene los submódulos de mail-sender que es el encargado del envío de correos, minio-client el cual se comunica con nuestro servidor de archivos (minio) y web-client que es el que realiza peticiones externas.



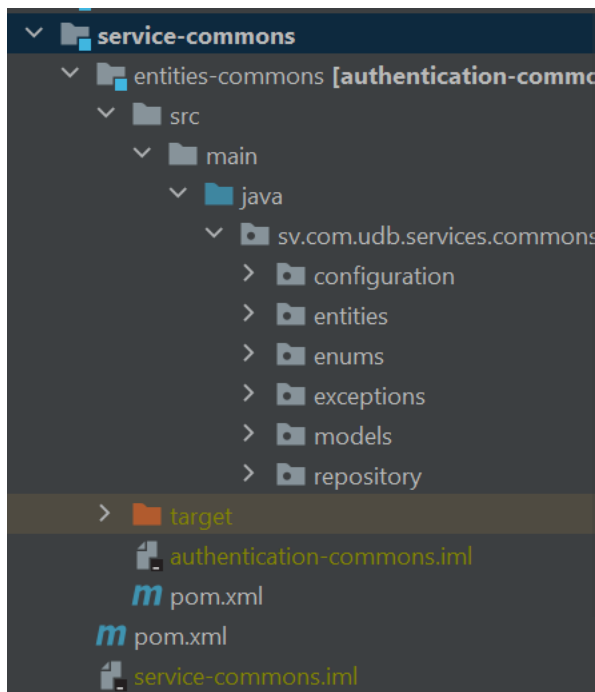
Docker-images

Contiene la configuración necesaria para poder crear contenedores Docker de cualquier modulo gracias a la ayuda del plugin yib



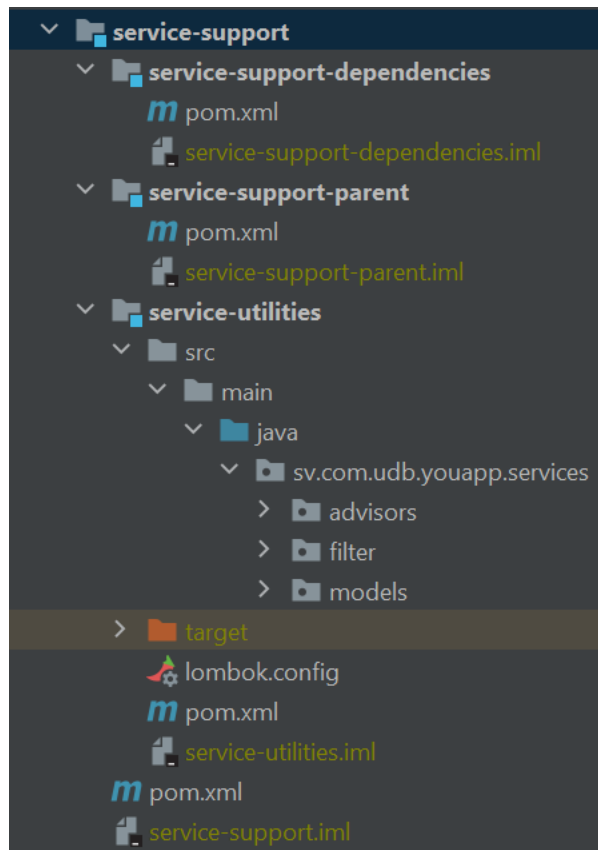
Service-commons

En este módulo se almacenan entidades, configuraciones, excepciones, repositorios, entre otros elementos que se compartirán entre los módulos que los necesiten, estas clases e interfaces se tienen en este módulo “común” para evitar la redundancia de las mismas.



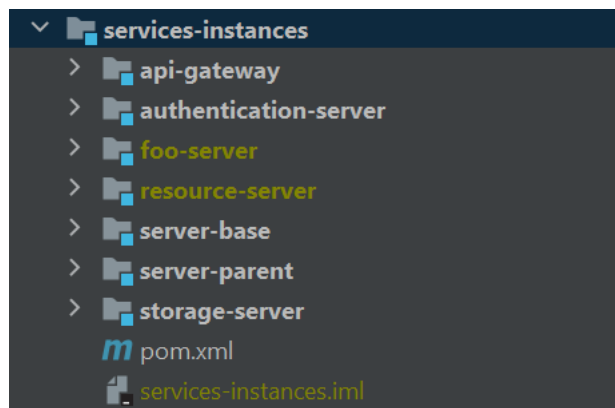
Services-support

Aquí almacenamos los módulos service-support-dependencies y service-support-parent los que en pocas palabras podemos decir que contienen nuestros bom; por otra parte tenemos el modulo service-utilities en el cual se tienen excepciones personalizadas como utilidad para el proyecto en general



Services-instances

Podemos definir este modulo como el contenedor de nuestros módulos finales, los cuales son: api-gateway, este se encarga de la redirección para nuestros endpoints para los microservicios authentication-server y storage-server; authentication-server, el cual se encarga del logeo usando OAuth2; foo-server y resource-server, los cuales contienen configuraciones de la seguridad; server-base y server-parent, que son los que contienen los pom.xml bases para los módulos funcionales tales como authentication-server y storage-server; storage-server, aquí se encuentra la solución a los requisitos funcionales de nuestra aplicación como la subida de canciones, creación y modificación de playlist, búsqueda de canciones, entre otros.



AuthenticationApplication.java

Es la clase que contiene el main para poder correr nuestro modulo authentication-server

```
package sv.com.udb.services.authentication;

import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@EntityScan("sv.com.udb.services")
@EnableJpaRepositories("sv.com.udb.services")
@SpringBootApplication(scanBasePackages = "sv.com")
public class AuthenticationApplication {
    public static void main(String[] args) {
        new SpringApplicationBuilder(AuthenticationApplication.class)
            .registerShutdownHook(true).run(args);
    }
}
```

StorageApplication.java

Es la clase que contiene el main para poder correr nuestro modulo storage-server

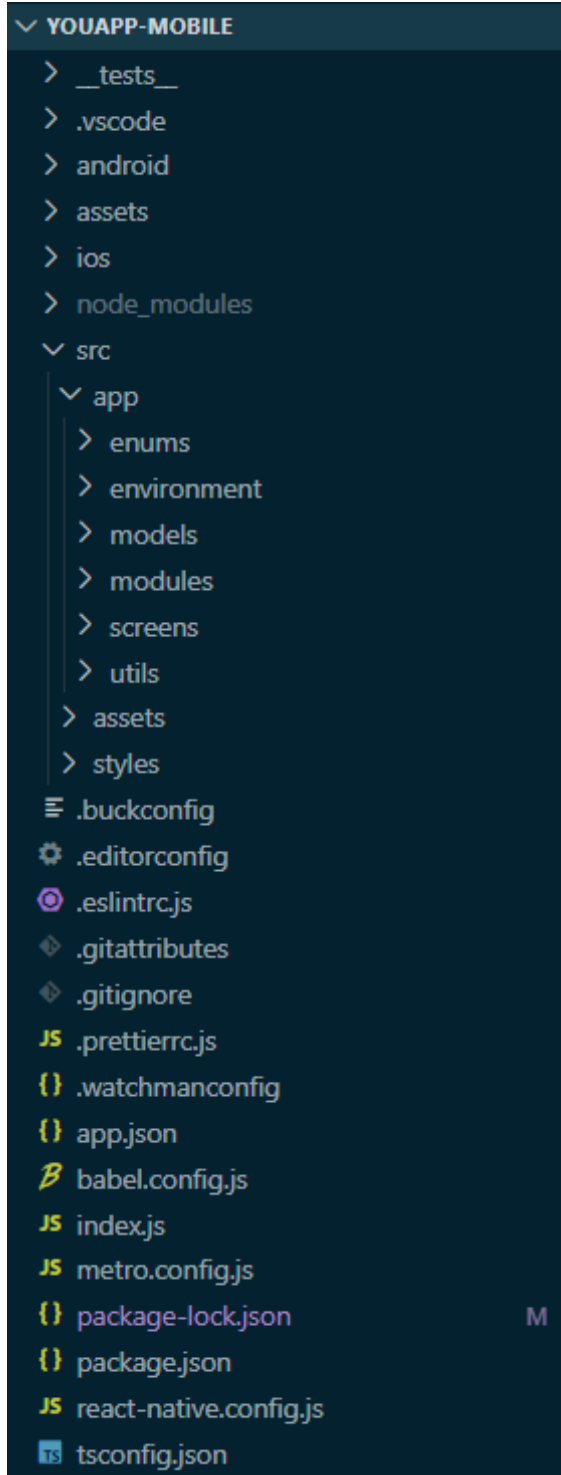
```
package sv.com.udb.youapp.services.storage;

import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@EntityScan("sv.com.udb.services")
@EnableJpaRepositories("sv.com.udb.services")
@SpringBootApplication(scanBasePackages = "sv.com")
public class StorageApplication {
    public static void main(String[] args) {
        new SpringApplicationBuilder(StorageApplication.class)
            .registerShutdownHook(true).run(args);
    }
}
```

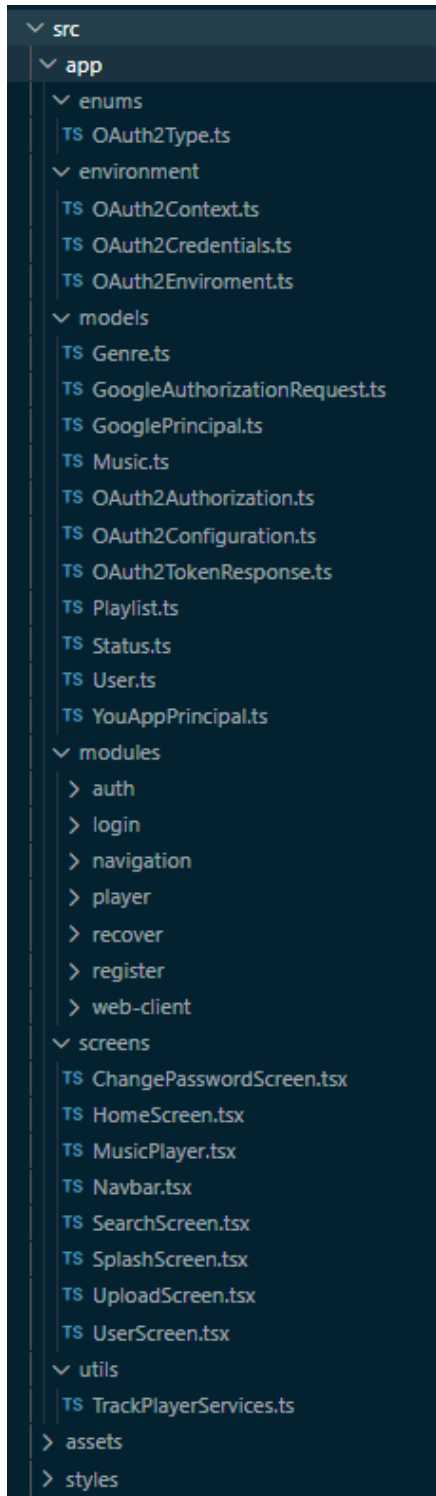
En ambos casos escaneamos las entidades con `@EntityScan` y habilitamos el uso de los repositorios con `@EnableJpaRepositories` por medio de la dirección de su paquete, también escaneamos los paquetes que podrá usar nuestro servicio usando la propiedad `scanBasePackages` dentro de la anotación `@SpringBootApplication`

Estructura de proyecto frontend



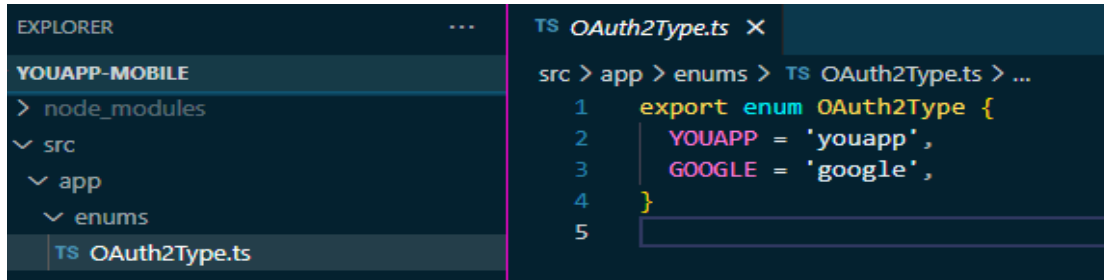
src

Contiene los archivos fuentes para el funcionamiento de nuestra aplicación



Enums

Contiene un archivo de utilidad tipo enum con las opciones de autenticación que posee YouApp

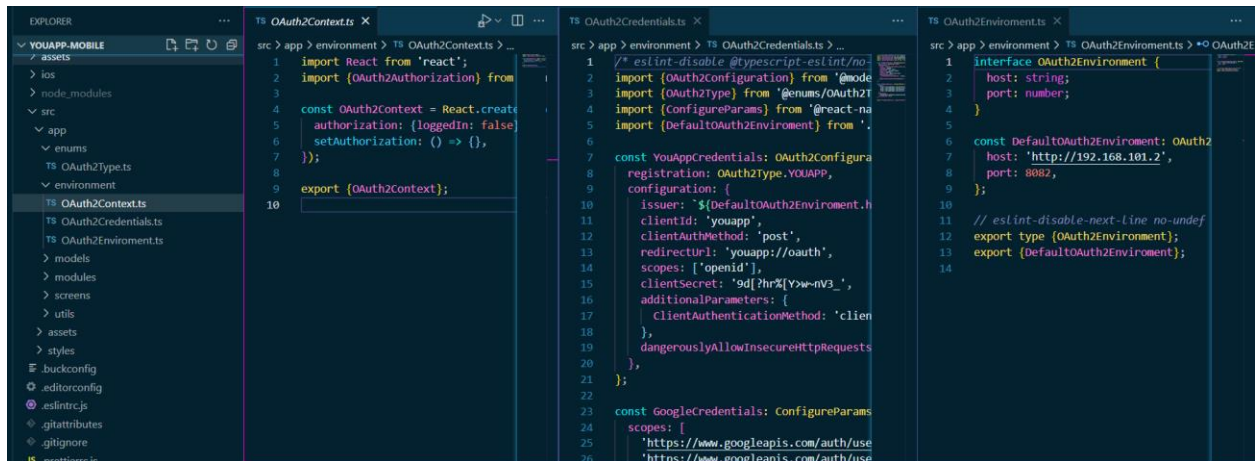


The screenshot shows the VS Code interface. On the left, the Explorer sidebar displays the project structure for 'YOUAPP-MOBILE', with the file 'TS OAuth2Type.ts' selected under the 'enums' directory. The main editor area shows the content of 'TS OAuth2Type.ts', which defines an enum for OAuth2 authentication types.

```
1 export enum OAuth2Type {
2   YOUAPP = 'youapp',
3   GOOGLE = 'google',
4 }
5
```

Environment

Como su nombre lo indica aquí tenemos las variables de nuestro entorno, en este caso la constante que expone la configuración necesaria para la seguridad con OAuth2, el host y el puerto para OAuth2, entre otros



The screenshot shows three open TypeScript files in VS Code. The Explorer sidebar on the left shows the project structure with 'TS OAuth2Context.ts', 'TS OAuth2Credentials.ts', and 'TS OAuth2Environment.ts' selected. The main editor area displays the code for these files.

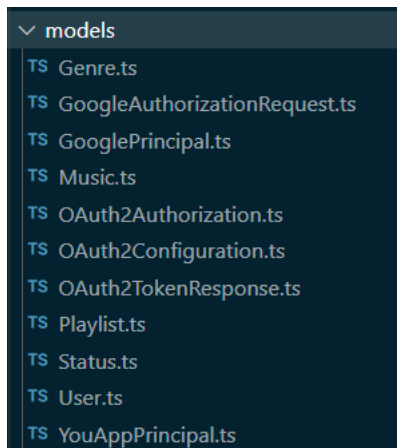
```
1 import React from 'react';
2 import {OAuth2Authorization} from 'react-native-oauth2';
3
4 const OAuth2Context = React.createContext({
5   authorization: {loggedIn: false},
6   setAuthorization: () => {},
7 });
8
9 export {OAuth2Context};
10
```

```
1 /* eslint-disable @typescript-eslint/no-explicit-any */
2 import {OAuth2Configuration} from '@react-native-community/oauth2';
3 import {OAuth2Type} from '@enums/OAuth2Type';
4 import {ConfigureParams} from 'react-native-oauth2';
5 import {DefaultOAuth2Environment} from './OAuth2Environment';
6
7 const YouAppCredentials: OAuth2Configuration = {
8   registration: OAuth2Type.YOUAPP,
9   configuration: {
10     issuer: `${DefaultOAuth2Environment.host}/oauth2`,
11     clientId: 'youapp',
12     clientAuthMethod: 'post',
13     redirectUrl: 'youapp://oauth2',
14     scopes: ['openid'],
15     clientSecret: '9d1?hr%{Yw-nv3_',
16     additionalParameters: {
17       clientAuthenticationMethod: 'client_secret_post',
18     },
19     dangerouslyAllowInsecureHttpRequests: true,
20   },
21 };
22
23 const GoogleCredentials: ConfigureParams = {
24   scopes: [
25     'https://www.googleapis.com/auth/userinfo.email',
26     'https://www.googleapis.com/auth/userinfo.profile',
27   ],
28 };
29
```

```
1 interface OAuth2Environment {
2   host: string;
3   port: number;
4 }
5
6 const DefaultOAuth2Environment: OAuth2Environment = {
7   host: 'http://192.168.101.2',
8   port: 8082,
9 };
10
11 // eslint-disable-next-line no-undef
12 export type {OAuth2Environment};
13 export {DefaultOAuth2Environment};
14
```

Models

Contiene las entidades a usar para el desarrollo de la aplicación

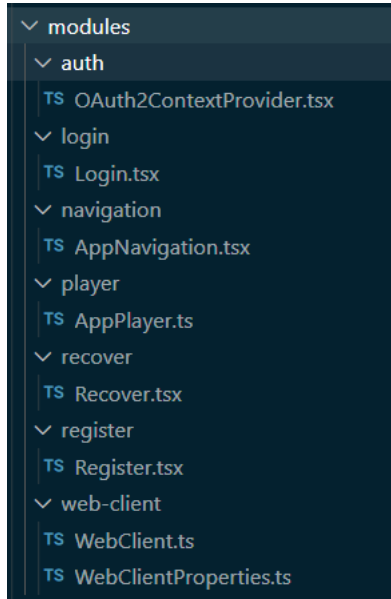


The screenshot shows the Explorer sidebar in VS Code with the 'models' directory expanded. It lists several TypeScript files used for data models in the application.

- TS Genre.ts
- TS GoogleAuthorizationRequest.ts
- TS GooglePrincipal.ts
- TS Music.ts
- TS OAuth2Authorization.ts
- TS OAuth2Configuration.ts
- TS OAuth2TokenResponse.ts
- TS Playlist.ts
- TS Status.ts
- TS User.ts
- TS YouAppPrincipal.ts

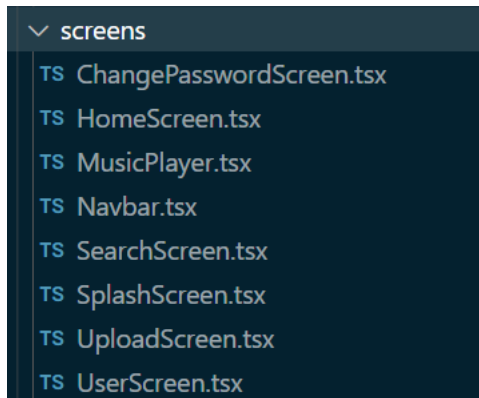
Modules

En esta carpeta tenemos los módulos que componen las pantallas de nuestra aplicación y también el módulo web-client el cual expone una instancia a WebClient para realizar peticiones y sus correspondientes propiedades



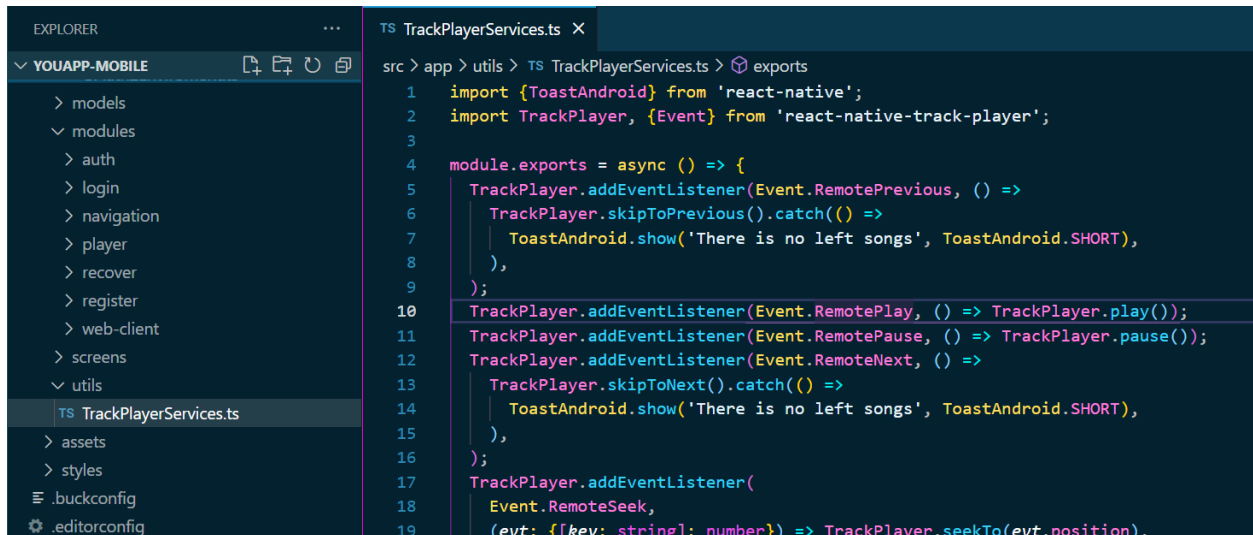
Screens

En esta carpeta se encuentran los componentes que componen las pantallas de la aplicación, valga la redundancia, o componentes genéricos.



Utils

En los archivos “útiles” del proyecto se tiene el servicio que nos permite tener el control de la reproducción de música

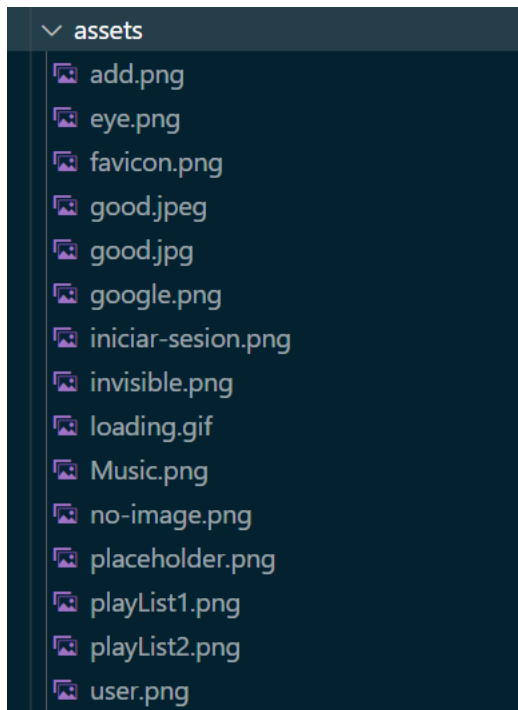


The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the project structure under 'YOUAPP-MOBILE'. The 'utils' directory is expanded, and 'TrackPlayerServices.ts' is selected. The main editor shows the code for this file, which exports a service for controlling a music player. The code includes imports for 'ToastAndroid' and 'TrackPlayer' from 'react-native' and 'react-native-track-player' respectively. It then defines a module with several event listeners for 'Event.RemotePrevious', 'Event.RemotePlay', 'Event.RemotePause', 'Event.RemoteNext', and 'Event.RemoteSeek', each with corresponding actions like 'skipToPrevious', 'play', 'pause', 'skipToNext', and 'seekTo'.

```
src > app > utils > TS TrackPlayerServices.ts > exports
1  import {ToastAndroid} from 'react-native';
2  import TrackPlayer, {Event} from 'react-native-track-player';
3
4  module.exports = async () => {
5      TrackPlayer.addEventListener(Event.RemotePrevious, () =>
6          TrackPlayer.skipToPrevious().catch(() =>
7              ToastAndroid.show('There is no left songs', ToastAndroid.SHORT),
8          ),
9      );
10     TrackPlayer.addEventListener(Event.RemotePlay, () => TrackPlayer.play());
11     TrackPlayer.addEventListener(Event.RemotePause, () => TrackPlayer.pause());
12     TrackPlayer.addEventListener(Event.RemoteNext, () =>
13         TrackPlayer.skipToNext().catch(() =>
14             ToastAndroid.show('There is no left songs', ToastAndroid.SHORT),
15         ),
16     );
17     TrackPlayer.addEventListener(
18         Event.RemoteSeek,
19         (evt: {[key: string]: number}) => TrackPlayer.seekTo(evt.position).
```

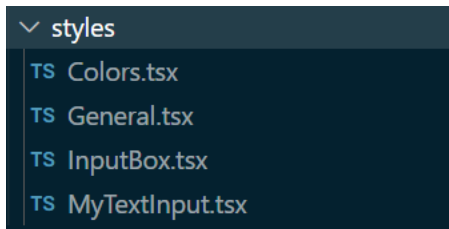
Assets

En esta carpeta se almacenan los archivos para estilizar la aplicación, tales como fuentes, imágenes, iconos, entre otros.



Styles

Aquí tenemos nuestros estilos y componentes estilizados para poder reutilizarlos.



Index.js

Esta es la clase principal para correr el proyecto react, la cual solo llama al componente que controla la navegación (El que gestiona las pantallas) de la aplicación y registra el servicio de la utilería como el servicio de sincronización con nuestro reproductor.

