import 'dart:async';

import 'package:bloc/bloc.dart';

import 'package:drawer\_shop/models/models.dart';

import 'package:drawer\_shop/models/wishlist\_model.dart';

import 'package:equatable/equatable.dart';

part 'wishlist\_event.dart';

part 'wishlist\_state.dart';

class WishlistBloc extends Bloc<WishlistEvent, WishlistState> {

  WishlistBloc() : super(WishlistLoading());

  @override

  Stream<WishlistState> mapEventToState(

    WishlistEvent event,

  ) async\* {

    if (event is StartWishlist) {

      yield\* \_mapStartWishlistToState();

    }

    else if (event is AddWishlistProduct) {

      yield\* \_mapAddWishlistProductToState(event, state);

    }

    else if (event is RemoveWishlistProduct) {

      yield\* \_mapRemoveWishlistProductToState(event, state);

  }

  }

       Stream<WishlistState> \_mapStartWishlistToState() async\*{

     yield WishlistLoading();

     try {

      await Future<void>.delayed(Duration(seconds: 1));

     yield const WishlistLoaded();

 ////////   emit(const WishlistLoaded(wishlist: Wishlist(products: []);)

  } catch (\_) {}

 }

    Stream<WishlistState> \_mapAddWishlistProductToState(

     AddWishlistProduct event,

      WishlistState state,

    ) async\*{

      if (state is WishlistLoaded) {

      try {

        yield WishlistLoaded(wishlist: Wishlist(products: List.from(state.wishlist.products)..add(event.product),),);

    } catch (\_) {}

  }

}

    Stream<WishlistState> \_mapRemoveWishlistProductToState(

     RemoveWishlistProduct event,

     WishlistState state,

    ) async\*{

      if (state is WishlistLoaded) {

      try {

        yield WishlistLoaded(wishlist: Wishlist(products: List.from(state.wishlist.products)..remove(event.product),),);

    } catch (\_) {}

  }

}

}