class User{

    constructor(name, password, email){

        this.name = name;

        this.password = password;

        this.email = email;

    }

}

class UserManager{

    constructor(){

    }

    users = [new User("Stoyan", "123ghj", "stoyan112@gmail.com")];//zashto pri opit da login-na Stoyan, razbiram, che toj veche e lognat???

    //loggedIn = null; //for one user

    loggedIn =[]; // for multiple users

    addUser(user){

        const userExist = this.users.filter(u => u.name === user.name);// return the user if he/she exist

        //console.log(userExist)

        if(userExist.length > 0){ // if so, from filter check we have element in the array, so the user EXIST

            console.log(`User with username ${user.name} already exist`);

        }else{

            this.users.push(user);

        }

    }

    login(user){

        //if(this.loggedIn){// if true, the user is logged in

        //console.log(user)

        //if(this.loggedIn.length > 0){ // works for only one logged in user

        if(this.loggedIn.includes(user)){ //works for multyple logged users

            console.log(`The user ${user.name} is already logged in`);

        }else{

            const userLogged = this.users.filter(u  =>

                u.name === user.name &&

                u.password === user.password);

            if(userLogged.length > 0){//if so, user with this un and pw exist

                //this.loggedIn = user; // works for only one user

                this.loggedIn.push(user) // works for multyple users

                //console.log(this.loggedIn)

            }else{

                console.log(`Wrong username or password`);

            }

        }

    }

    loggedUsers(){ // return logged in users;

        return console.log(this.loggedIn);

    }

    logout(user){ //logout user

        if(this.loggedIn.includes(user)){

            // this.loggedIn.splice(this.loggedIn.indexOf(user),1); // loggedIn is an array with logged in people, with splice i do remove them and they are no longer logged

            this.loggedIn = this.loggedIn.filter(element => element.name !== user.name);// both ways to logout works perfectly

        }else{

            console.log(`User ${user} is not logged in!`);

        }

    }

    deleteUser(user){ //delete user

        this.users = this.users.filter(element => element.name !== user.name);

    }

    existingUsers(){// list existing users in system

        return console.log(this.users);

    }

    changePassword(user, currentPassword, newPassword){

        const userExist = this.users.filter(u => u.name === user.name && u.password === user.password); //check if the user exist.... return the user as obj in array!

        const userLoggedIn = this.loggedIn.includes(user); // check if the user is logged in ..... returns true/false!

        const correctPassword = userExist[0].password === currentPassword; // check if the user's password is the same as the parametur...... returns true/false!

        if(userExist && userLoggedIn && correctPassword){

            let password = newPassword.split("");

            password = password.filter( el => el === el.toUpperCase()) // check if the password contains upper lettar, number or symbol

            //if so "password" will contain array with elements eles it would be empty array!

            if(newPassword.length >= 6 && password != []){

                userExist[0].password = newPassword;// da go oprawq parolata otiva v glavniq model!!!!, Ne, nqma kakvo da opravqm, obekta e referenten tip danni, toest novata parola si otiva w obekta!

            }else{

                console.log("New password does not match to the requirements: length 6 or bigger, number and symbol");

            }

        }

    }

}

class Vehicle{

    constructor(make, model, topSpeed, fuel){

        this.make = make;

        this.model = model;

        this.topSpeed = topSpeed;

        this.fuel = fuel;

    }

}

class Car extends Vehicle{

    constructor(make, model, topSpeed, fuel, isItSport){

        super(make, model, topSpeed, fuel);

        this.isItSport = isItSport;

    }

}

class Bike extends Vehicle{

    constructor(make, model, topSpeed,fuel, cC, ){

        super(make, model, topSpeed, fuel);

        this.cC = cC;

    }

}

class Jeep extends Vehicle{

    constructor(make, model, topSpeed, fuel, forByFor){

        super(make, model, topSpeed, fuel);

        this.forByFor = forByFor;

    }

}

class Offer{

    constructor(title, price, photos, description, vehicle, offerCreator){

        this.title = title;

        this.price = price;

        this.photos = photos;

        this.description = description;

        this.vehicle = vehicle;

        this.offerCreator = offerCreator;

    }

}

class OfferManager{

    offers = [];

    comands = ["title", "price","photos", "description","vehicle"];

    constructor(){

    }

    addOffer(offer, userManager) {

        userManager.loggedIn.forEach(element => {

            if(offer.offerCreator.name === element.name && // check if the creator is logged in

               !this.offers.includes(offer)){ // check does the offer includes in our offers list. If false it would be true, so the new offer will be added to the list!

                this.offers.push(offer);

            }

        });

    }

    deleteOffer(offer, userManager){

        const isCreatorLogged = userManager.loggedIn.includes(offer.offerCreator); // true/false depends if the user is logged in or not

        if(isCreatorLogged){

            this.offers = this.offers.filter(element => element !== offer); // delete offer from offers list

        }else{

            console.log("The offer cannot be deleted while user is not logged in!");

        }

    }

    editOffer(offer, userManager, comand, param){

        const isCreatorLogged = userManager.loggedIn.includes(offer.offerCreator);

        if(isCreatorLogged){

            if(this.comands.includes(comand)){

                this[comand](param); //one of the methods down (title,price, discription, vehicle).

            }else{

                console.log("Wrong comand!")

            }

        }

    }

    title(param){

        offer.title = param;

    }

    price(param){

        offer.price = param;

    }

    photos(param){

        offer.photos.pop()

        offer.photos.push(param);

    }

    description(param){

        offer.description = param;

    }

    vehicle(param){

        offer.vehicle = param;

    }

    // filterOffer(make, model, topSpeed, fuel, isItSport){

    filterOffer(){

        let reference = ["make", "model", "price", "fuel", "isItSport"]

        let resultObj = {};

        for(let i=0; i<arguments.length; i++){

            //make, model, price, fuel, isItSport

            resultObj[`${reference[i]}`] = arguments[i];

        }

        let result = this.offers.filter( function(offer){

            let make = offer.vehicle.make.toLowerCase() == resultObj.make.toLowerCase().trim();

            let model = offer.vehicle.model.toLowerCase() == resultObj.model.toLowerCase().trim();

            let price = Number(offer.price.trim()) <= Number(resultObj.price);//trim()?

            return make && model && price;

        });

        console.log("Tuk:", result);

    }

}

let bmw=new Car("Bmw", "318d", "245", "diesel", true);

let honda = new Car("Honda", "Civic 1.4i", "190", "petrol", false);

let honda2 = new Car("Honda", "Civic 1.6i", "200", "petrol", true);

//console.log(bmw)

//create user manager and offer manager

let userManager = new UserManager();

let offerManager = new OfferManager();

let chovek = new User("Pesho", "pesho123", "peshoSkupoto@gmail.com");

//add user to the system and login

userManager.addUser(chovek)

userManager.login(chovek)

//create an offer

let offer = new Offer("title", "24500 ",["snimki","Mnimki"], "Mnogo zapazena, karana ot baba", bmw, chovek)

let offer1 = new Offer("title", "12500 ",["snimki","Mnimki"], "Mnogo zapazena, karana ot baba", honda, chovek);

let offer2 = new Offer("title", "14500 ",["snimki","Mnimki"], "Mnogo zapazena, karana ot baba", honda2, chovek)

let p = new User("Pima", "111", "pima@gmail.com")

//add user test

userManager.addUser(p)// add user to users

//login user test

userManager.login(p) // login user

//userManager.loggedUsers();// list logged in users!

//logout user test

userManager.logout(p); // logout user

userManager.deleteUser(p)

userManager.existingUsers()

// delete user test

// userManager.deleteUser(chovek); // delete user;

// userManager.existingUsers()

//logged in users test

//console.log("loggedin")

//userManager.loggedUsers()

//add offer test

offerManager.addOffer(offer, userManager)

offerManager.addOffer(offer1, userManager)

offerManager.addOffer(offer2, userManager)

//detete offer test

//offerManager.deleteOffer(offer,userManager)

//edit offer test

//offerManager.editOffer(offer, userManager, "vehicle", honda);

//console.log(offerManager.offers)

//filter offer test

//offerManager.filterOffer("Bmw", "116d", "14800,00");//make, model, price, fuel, isItSport

//offerManager.filterOffer("Bmw", "318d","33000");

//change password test

// chovek -> user, current password, new password

userManager.changePassword(chovek, "pesho123","Pesho123!");

userManager.existingUsers()

console.log(chovek)