НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №6

з дисципліни

“Програмування мобільних систем / Розроблення клієнтських додатків для мобільних платформ”

Виконав:

студент групи IO-83

ЗК 8321

Морозов Андрій

Київ 2021

**Лістинг коду**

App.js

import React from 'react';

import RootNavigator from './screens/RootNavigator'

const App = () => {

return (

<RootNavigator />

);

}

export default App

MainTab.js

import React from "react";

import { View, Text } from 'react-native'

const MainTab = () => {

return (

<View style={{

flex: 1,

justifyContent: 'center',

flexDirection:'column',

alignItems:'center',

}}>

<Text style={{ fontSize: 20 }}>Морозов Андрій</Text>

<Text style={{ fontSize: 20 }}>Група ІО-83</Text>

<Text style={{ fontSize: 20 }}>ЗК ІО-8321</Text>

</View>

)

}

export default MainTab

SecTab.js

import React, { useState, useEffect } from 'react';

import {StyleSheet, Text, View, Switch, Dimensions} from 'react-native';

import {LineChart, PieChart} from "react-native-chart-kit";

import { data , labels, useScreenDimensions } from '../consts/consts'

import Svg, { Circle } from 'react-native-svg';

const Graphs = ({ navigation }) => {

const dim = Dimensions.get("screen")

const [isEnabled, setIsEnabled] = useState(false);

const toggleSwitch = () => setIsEnabled(previousState => !previousState);

const checkOrientation = () => {

const dim = Dimensions.get('screen');

if (dim.height >= dim.width) {

return portrait

} else {

return landscape

}

}

const screenData = useScreenDimensions();

if (isEnabled) {

return (

<View style={checkOrientation().container}>

<Text>Line Chart</Text>

<Switch

trackColor={{ false: "#BE7B4A", true: "#4169e1" }}

thumbColor={isEnabled ? "#BE7B4A" : "#4169e1"}

ios\_backgroundColor="#BE7B4A"

onValueChange={toggleSwitch}

value={isEnabled}

style={checkOrientation().toggle}

/>

<PieChart

data={[

{

percent: 5,

color: '#654321',

},

{

percent: 5,

color: '#0cb4ff',

},

{

percent: 10,

color: '#BE7B4A',

},

{

percent: 80,

color: '#4169e1',

},

]}

hasLegend={false}

width={

screenData.isLandscape ?

Dimensions.get('screen').width :

Dimensions.get('screen').width

}

height={

screenData.isLandscape ?

Dimensions.get("screen").height / 1.8:

Dimensions.get("screen").height / 3

}

chartConfig={{

color: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

}}

style={{

alignItems: "center",

marginLeft: '50%',

}}

accessor="percent"

absolute

/>

<View style={

screenData.isLandscape ?

{ zIndex: 1, position: 'absolute', paddingLeft: '39.5%', top: '43.7%' } :

{ zIndex: 1, position: 'absolute', paddingLeft: '35.5%', top: '51.7%' }

}>

<Svg height="180" width="200">

<Circle cx="50" cy="50" r="50" fill="white" />

</Svg>

</View>

</View>

)

} else {

return (

<View style={checkOrientation().container}>

<Text>Pie</Text>

<Switch

trackColor={{ false: "#4169e1", true: "#D4B057" }}

thumbColor={isEnabled ? "#BE7B4A" : "#D4B057"}

ios\_backgroundColor="#4169e1"

onValueChange={toggleSwitch}

value={isEnabled}

style={checkOrientation().toggle}

/>

<LineChart

data={{ labels: labels, datasets: [{ data: data }] }}

width={ screenData.isLandscape ? dim.width : dim.width \* 1.3 }

height={ screenData.isLandscape ? dim.height / 4.5 : dim.height / 6 }

chartConfig={{

backgroundColor: "rgb(242, 242, 242)",

backgroundGradientFrom: "rgb(242, 242, 242)",

backgroundGradientTo: "rgb(242, 242, 242)",

color: (opacity = 1) => `rgba(0, 0, 0, ${opacity})`,

propsForDots: {

r: "0",

strokeWidth: "0",

stroke: "#000",

barPercentage: '1'

}

}}

style={

screenData.isLandscape ?

{

paddingRight: dim.width / 4.5,

marginLeft: dim.width / 10,

marginTop: dim.height / 6.5,

} :

{

paddingRight: dim.width / 4,

marginLeft: dim.width / 9,

marginTop: dim.height / 11,

}

}

withInnerLines={false}

withOuterLines={false}

withHorizontalLabels={false}

withVerticalLabels={false}

bezier

/>

</View>

)

}

}

export default Graphs

const portrait = StyleSheet.create({

container: {

flex: 0,

alignItems: "center",

justifyContent: "center",

marginTop: '30%',

},

toggle: {

flex: 0,

marginTop: 5,

justifyContent: 'center',

alignItems: 'center',

marginBottom: '15%'

},

});

const landscape = StyleSheet.create({

container: {

flex: 0,

alignItems: "center",

justifyContent: "center",

marginTop: '5%',

},

toggle: {

flex: 0,

marginTop: 5,

justifyContent: 'center',

alignItems: 'center',

marginBottom: 10,

},

});

RootNavigator.js

import React from 'react';

import { View } from 'react-native';

import Icon from 'react-native-vector-icons/FontAwesome'

import { createMaterialBottomTabNavigator } from '@react-navigation/material-bottom-tabs';

import { NavigationContainer } from '@react-navigation/native';

import { MainBtmBarTheme } from '../consts/consts'

import MainTab from "./MainTab";

import SecTab from "./SecTab";

import Books from './Books'

const Tab = createMaterialBottomTabNavigator();

const RootNavigator = () => {

return (

<NavigationContainer theme={ MainBtmBarTheme }>

<Tab.Navigator

shifting={true}

sceneAnimationEnabled={true}

initialRouteName="Creator"

>

<Tab.Screen

name="MainTab"

component={MainTab}

options={{

tabBarLabel: 'MainTab',

tabBarIcon: () => (

<View>

<Icon

style={[{color: '#F9F3E7'}]}

size={25}

name={'user'}

/>

</View>

),

}}

/>

<Tab.Screen

name="SecTab"

component={SecTab}

options={{

tabBarLabel: 'SecTab',

tabBarIcon: () => (

<View>

<Icon

style={[{color: '#F9F3E7'}]}

size={25}

name={'asterisk'}

/>

</View>

),

}}

/>

<Tab.Screen

name="Books"

component={Books}

options={{

tabBarLabel: 'Books',

tabBarIcon: () => (

<View>

<Icon

style={[{color: '#F9F3E7'}]}

size={25}

name={'bold'}

/>

</View>

),

}}

/>

</Tab.Navigator>

</NavigationContainer>

);

}

export default RootNavigator

Books.js

import React, { useState } from "react";

import { View, Text, ScrollView, Image, TouchableHighlight, Dimensions} from 'react-native'

import BooksList from '../assets/BooksList.json'

import {

useScreenDimensions, getImageFromLocalData,

HeaderBarTheme, getInfoFromLocalData

} from '../consts/consts'

import { Appbar } from 'react-native-paper';

import Icon from 'react-native-vector-icons/FontAwesome'

import SearchBar from "react-native-dynamic-search-bar";

const Books = ({ navigation }) => {

const dim = Dimensions.get('screen')

const [searchBarText, setSearchBarText] = useState('')

const [booksData, setBooksData] = useState([])

const screenData = useScreenDimensions();

const chk = screenData.isLandscape

const filterBooks = (Data, text) => {

if(text.trim().length === 0 || text.length === 0) {

return Data

} else {

return Data.filter((item) => {

if( item.title.replace(/[^a-zA-Z ]/g, "").toLowerCase().indexOf(text)> -1 ){

return (item)

}

})

}

}

const currentData = filterBooks(booksData, searchBarText)

const delFromArray = (id) => {

const idx = booksData.findIndex((el) => el.isbn13 === id)

const newBooksData = [...booksData.slice(0, idx),...booksData.slice(idx + 1)]

setBooksData(newBooksData)

};

const deleteDpl = (arr, key) => {

return [...new Map(arr.map(item => [item[key], item])).values()]

}

const UpdatedDataOfBooks = async (text) => {

let BooksArrayFromData = []

const txt = text.toLowerCase().replace(/[^a-zA-Z ]/g, "").replace(/\s+/g, ' ').trim().replace(/,/g, '')

setSearchBarText(txt)

if( txt.length <= 2) {

return null

} else if ( txt.length >= 2 ) {

let url = `https://api.itbook.store/1.0/search/${txt}`

const fetchResult = await fetch(url);

const loadedData = await fetchResult.json();

BooksArrayFromData = [

...filterBooks(booksData, txt),

...loadedData.books

]

}

setBooksData(deleteDpl([...BooksArrayFromData, ...booksData], 'isbn13' ))

}

return (

<ScrollView style={{ backgroundColor: '#f8ecdd' }}>

<View>

<Appbar.Header theme={ HeaderBarTheme }>

<Appbar.Action

icon="home"

/>

<SearchBar

style={{ backgroundColor: '#f8ecdd', flex: 1}}

placeholder="Search"

onClearPress={() => {setSearchBarText('')}}

onChangeText={

(text) => UpdatedDataOfBooks(text)

}

/>

<Appbar.Action

icon="plus"

onPress={() => {

navigation.navigate('BkAdd', {

booksData: booksData,

setBooksData: setBooksData

});

}}

/>

</Appbar.Header>

</View>

<View>

{

currentData.length === 0 ?

<View style={{

height: dim.height,

paddingTop: screenData.isLandscape ? '15%' : '65%',

flexDirection:'column',

alignItems:'center'

}}>

<Text style={{fontSize: 20}}>

No items

</Text>

</View> :

searchBarText.length <= 2 ?

<View style={{

height: dim.height,

paddingTop: screenData.isLandscape ? '15%' : '65%',

flexDirection:'column',

alignItems:'center'

}}>

<Text style={{fontSize: 20}}>

There are no such books

</Text>

</View> :

currentData.map((item, i) => {

return(

<View key={i}>

<TouchableHighlight

onPress={() => {

navigation.navigate('BkInfo', {

Id: item.isbn13,

});

}}

>

<View style={{

backgroundColor: '#488962',

borderRadius: 30,

flexDirection: 'row',

margin: 10

}}>

<View>

<Image

resizeMode="cover"

source={

item.image === 'N/A' ?

require('../assets/coming\_soon.jpeg') :

{uri: item.image}

}

style={{

borderRadius: 30,

height: 200,

width: 150

}}

/>

</View>

<View style={{

marginLeft: '5%',

width: '76%'

}}>

<Text style={{

flex: 0,

width: chk ? '100%' : '45%',

fontSize: 18,

marginBottom: 10,

marginTop: 10,

textAlign: 'left',

}}>

{

item.title.length >= 43 ?

item.title.slice(0, 43 - 1) + '…' :

item.title

}

</Text>

<Text style={{

flex: 0,

width: chk ? '100%' : '45%',

fontSize: 15,

marginBottom: 10,

marginTop: 10,

textAlign: 'left',

}}>

{

item.subtitle.length === 0 ?

'Programming skills' :

item.subtitle.length >= 40 ?

item.subtitle.slice(0, 40 - 1) + '…' :

item.subtitle

}

</Text>

<Text style={{

position: 'absolute',

bottom: -15,

marginBottom: '5%'

}}>

Price: {

item.price.length === 0 ?

'$100' :

item.price

}

</Text>

</View>

<TouchableHighlight

style={{

position: "absolute",

right: 0,

width: chk ? '8%' : '12%',

height: '100%',

borderRadius: chk ? 25 : 30,

backgroundColor: '#A0C265'

}}

onPress={() => { delFromArray(item.isbn13) }}>

<View>

<Icon

onPress={() => { delFromArray(item.isbn13) }}

style={[{

color: '#675649',

flex: 0,

marginTop: chk ? '145%' : '180%',

alignSelf: 'center',

}]}

size={25}

name={'arrow-down'}

/>

</View>

</TouchableHighlight>

</View>

</TouchableHighlight>

</View>

)

})

}

</View>

</ScrollView>

)

}

export default Books

BkInfo.js

import React, { useEffect, useState } from "react";

import { getImageFromLocalData, getInfoFromLocalData } from "../consts/consts";

import {

View, Text,

StyleSheet, Dimensions, ScrollView, TouchableHighlight, Image

} from 'react-native'

const BkInfo = ({ route }) => {

const { Id } = route.params;

const [fullArray, setFullArray] = useState([])

useEffect(() => {

let cleanupFunction = false;

const fetchData = async () => {

try {

fetch(`https://api.itbook.store/1.0/books/${Id}`)

.then(response => response.json() )

.then( data => setFullArray([data]))

if(!cleanupFunction) {

setFullArray([]);

}

} catch (e) {

console.error(e.message)

}

};

fetchData();

return () => cleanupFunction = true;

}, []);

return (

<ScrollView style={{backgroundColor: '#f8ecdd'}}>

<View>

<View style={{ flex: 0, alignItems: 'center', justifyContent: 'center' }}>

{

fullArray.map((item, index) => {

return (

<View key={index}>

<View style={orientation().mainTopContainer}>

<TouchableHighlight style={orientation().imgContainer}>

<Image

resizeMode="cover"

source={

item.image === 'N/A' ?

require('../assets/coming\_soon.jpeg') :

{uri: item.image}

}

style={orientation().img}

/>

</TouchableHighlight>

<View style={orientation().topRightContainer}>

<Text style={orientation().txtTitle}>

{item.title}

</Text>

<View>

<Text

style={orientation().topTxt}>

{item.subtitle}

</Text>

</View>

<View>

<Text

style={orientation().topTxt}>

Year - {item.year}

</Text>

</View>

<View>

<Text

style={orientation().topTxt}>

Price - ${item.price}

</Text>

</View>

<View>

<Text

style={orientation().topTxt}>

Pages - {item.pages}

</Text>

</View>

</View>

</View>

<View style={orientation().bottomTxtContainer}>

<Text style={orientation().bottomTitleTxt}>Authors</Text>

<Text style={orientation().bottomTxt}>{item.authors}</Text>

<Text style={orientation().bottomTitleTxt}>Publisher</Text>

<Text style={orientation().bottomTxt}>{item.publisher}</Text>

<Text style={orientation().bottomTitleTxt}>Rating</Text>

<Text style={orientation().bottomTxt}>{item.rating}</Text>

<Text style={orientation().bottomTitleTxt}>Description</Text>

<Text style={orientation().bottomTxt}>{item.desc}</Text>

</View>

</View>

)

})

}

</View>

</View>

</ScrollView>

)

}

const portrait\_styles = StyleSheet.create({

mainTopContainer: {

flex: 1,

flexDirection: 'row',

flexWrap: 'wrap',

alignItems: "center",

},

topRightContainer: {

width: 180

},

imgContainer: {

marginLeft: '2%',

marginTop: 25,

height: 255,

width: 155,

},

img: {

height: 245,

width: 145,

borderRadius: 20,

},

txtTitle: {

width: 200,

fontSize: 22,

marginBottom: 5,

fontWeight: 'bold'

},

topTxt: {

color: '#292929',

fontSize: 18,

marginBottom: 5,

},

bottomTxtContainer: {

marginLeft: 10,

marginRight: 10

},

bottomTitleTxt: {

color: '#292929',

fontSize: 20,

marginBottom: 5,

fontWeight: 'bold'

},

bottomTxt:{

color: '#292929',

fontSize: 18,

marginBottom: 5,

},

});

const landscape\_styles = StyleSheet.create({

mainTopContainer: {

flex: 1,

flexDirection: 'row',

flexWrap: 'wrap',

alignItems: "center",

},

topRightContainer: {

width: 180,

},

imgContainer: {

marginLeft: '1%',

marginTop: 25,

height: 255,

width: 155,

},

img: {

height: 245,

width: 145,

borderRadius: 20,

},

txtTitle: {

width: 230,

fontSize: 22,

marginBottom: 5,

fontWeight: 'bold'

},

topTxt: {

color: '#292929',

width: 190,

fontSize: 18,

marginBottom: 5,

},

bottomTxtContainer: {

marginLeft: 10,

marginRight: 10

},

bottomTitleTxt: {

color: '#292929',

fontSize: 20,

marginBottom: 5,

fontWeight: 'bold'

},

bottomTxt:{

color: '#292929',

fontSize: 18,

marginBottom: 5,

},

});

const orientation = () => {

const dim = Dimensions.get('screen');

if (dim.height >= dim.width) {

return portrait\_styles

} else {

return landscape\_styles

}

}

export default BkInfo

BkAdd.js

import React, { useState } from "react";

import { View, Text, StyleSheet, ScrollView, TouchableOpacity } from 'react-native'

import { BtnMainTheme } from "../consts/consts";

import { Button } from "react-native-elements";

import { FloatingLabelInput } from 'react-native-floating-label-input';

const AddScreen = ({ navigation, route }) => {

const { booksData } = route.params;

const { setBooksData } = route.params;

const [title, setTitle] = useState('');

const [subtitle, setSubtitle] = useState('');

const [price, setPrice] = useState('');

function ValidationCallback(arr, setArr) {

try {

const num = parseInt(price);

let Idx = 50

if (isNaN(num)) {

setPrice('U entered letters to price')

setTimeout(() => {

setPrice('')

}, 2000);

} else {

const newItem = {

image: '',

isbn13: Idx,

price: price,

subtitle: subtitle,

title: title,

}

const newBooksData = [...arr, newItem]

setArr(newBooksData)

navigation.navigate('Books')

}

}catch(error) {

console.log('error', error);

}

}

return (

<ScrollView

keyboardShouldPersistTaps="handled"

>

<View style={{padding: 50, flex: 1}}>

<View style={{marginBottom: 20}}>

<FloatingLabelInput

countdownLabel="chars left"

placeholder={''}

maxLength={100}

showCountdown={true}

style={{color: '#fff'}}

label={'Title'}

value={title}

rightComponent={(

<TouchableOpacity

style={{ alignContent:'center', justifyContent:'center' }}

onPress={()=>{

setTitle('')

}}>

<Text>✕</Text>

</TouchableOpacity>

)}

onChangeText={(val) => setTitle(val)}

/>

</View>

<View style={{marginBottom: 20}}>

<FloatingLabelInput

label={'Subtitle'}

value={subtitle}

rightComponent={(

<TouchableOpacity

style={{

alignContent:'center',

justifyContent:'center'

}}

onPress={()=>{

setSubtitle('')

}}>

<Text>✕</Text>

</TouchableOpacity>

)}

onChangeText={(val) => setSubtitle(val)}

/>

</View>

<View style={{marginBottom: 20}}>

<FloatingLabelInput

keyboardType="numeric"

label={'Price'}

value={price}

rightComponent={(

<TouchableOpacity

style={{alignContent:'center', justifyContent:'center'}}

onPress={()=>{

setPrice('')

}}>

<Text>✕</Text>

</TouchableOpacity>

)}

onChangeText={(val) => setPrice(val)}

/>

</View>

<View style={{ flex: 1, alignItems: 'center' }}>

<Button

onPress={() => {

ValidationCallback(booksData, setBooksData)

}}

theme={BtnMainTheme}

title="Add"

buttonStyle={{ width: 150 }}

/>

</View>

</View>

</ScrollView>

)

}

export default AddScreen

ImageView.js

import React from 'react';

import { View, StyleSheet } from 'react-native';

import Image from 'react-native-image-progress';

const ImageView = ({ imagesArray, width, height}) => {

const normalImageSize = {

width: width,

height: height

}

const tripleImageSize = {

width: width \* 3,

height: height \* 3

};

const ImageOf = (uri, optionsStyles = normalImageSize) => (

<Image

style={optionsStyles}

source={uri}

threshold={150}

/>

);

return (

<View style={styles.imageBlockContainer}>

<View style={styles.smallImageContainer}>

{imagesArray[0] && ImageOf(imagesArray[0])}

{imagesArray[3] && ImageOf(imagesArray[1])}

{imagesArray[5] && ImageOf(imagesArray[2])}

</View>

{imagesArray[1] && ImageOf(imagesArray[0], tripleImageSize)}

<View style={styles.smallImageContainer}>

{imagesArray[2] && ImageOf(imagesArray[3])}

{imagesArray[4] && ImageOf(imagesArray[4])}

{imagesArray[6] && ImageOf(imagesArray[5])}

</View>

</View>

);

};

const styles = StyleSheet.create({

imageBlockContainer: {

display: "flex",

flexDirection: "row",

},

smallImageContainer: {

display: "flex",

flexDirection: "column"

}

})

export default ImageView

ImageLogic.js

import React, { useState, useEffect } from 'react';

import { View, Text, ScrollView, StatusBar, Dimensions, StyleSheet } from 'react-native';

import \* as ImagePicker from 'expo-image-picker';

import { Appbar } from 'react-native-paper';

import SearchBar from "react-native-dynamic-search-bar";

import ImageView from "./ImageView";

import { useScreenDimensions, HeaderBarTheme } from '../../consts/consts'

const ImageLogic = () => {

const [imageView, setImageView] = useState([]);

const pickImage = async () => {

const pickedImage = await ImagePicker.launchImageLibraryAsync({

mediaTypes: ImagePicker.MediaTypeOptions.Images,

allowsEditing: true,

aspect: [3, 4],

quality: 1

});

if (pickedImage.cancelled) {

console.log('Warn: cancelled')

} else {

setImageView(prevState => [...prevState, { uri: pickedImage.uri }])

}

};

useEffect(() => {

const url = `https://pixabay.com/api/?key=19193969-87191e5db266905fe8936d565&q=red+cars&image\_type=photo&per\_page=21`;

let cleanupFunction = false;

const fetchData = async () => {

try {

const fetchResult = await fetch(url);

const loadedData = await fetchResult.json();

const loadedDataURIs = loadedData['hits'].map((lD) => ({ uri: lD['largeImageURL'] }));

setImageView(loadedDataURIs)

} catch (e) {

console.error(e.message)

}

};

fetchData();

return () => cleanupFunction = true;

}, []);

const screenData = useScreenDimensions();

const galleryComponent = arraySubSplitter(imageView).map(

image => (

<ImageView

key={image[0].uri}

imagesArray={image}

width={screenData.width / 5}

height={

screenData.isLandscape ?

screenData.height / 2.5 :

screenData.height / 8

}

/>

)

);

return (

<>

<View>

<Appbar.Header theme={HeaderBarTheme}>

<Appbar.Action icon="home"/>

<SearchBar

placeholder='Search'

style={{flex: 1}}

/>

<Appbar.Action

icon="plus"

onPress={pickImage}

/>

</Appbar.Header>

</View>

<View style={orientation().galleryContainer}>

{

imageView.length !== 0 && (

<ScrollView style={orientation().imgContainer}>

{ galleryComponent }

</ScrollView>

)}

{

imageView.length === 0 &&

<View style={orientation().emptyView}>

<Text

style={{ fontStyle: "italic", fontSize: 20}}

>

No one

</Text>

</View>

}

</View>

</>

);

};

const arraySubSplitter = (arr = [], maxArrSize = 7) => {

const result = [];

for (let i = 0; i < Math.ceil(arr.length / maxArrSize); i++) {

result[i] = arr.slice(i \* maxArrSize, (i \* maxArrSize) + maxArrSize);

}

return result;

};

const portrait\_styles = StyleSheet.create({

imgContainer: {

display: "flex",

flexWrap: "wrap",

flexDirection: "row",

marginTop: 10,

},

galleryContainer: {

flex: 1,

marginTop: StatusBar.currentHeight

},

picker: {

alignSelf: "center",

width: "100%"

},

emptyView: {

flex: 1,

alignItems: "center",

justifyContent: "center",

height: "100%"

},

})

const landscape\_styles = StyleSheet.create({

imgContainer: {

display: "flex",

flexWrap: "wrap",

flexDirection: "row",

marginTop: 10,

},

galleryContainer: {

flex: 1,

marginTop: StatusBar.currentHeight

},

picker: {

alignSelf: "center",

width: "100%"

},

emptyView: {

flex: 1,

alignItems: "center",

justifyContent: "center",

height: "100%"

},

})

const orientation = () => {

const dimension = Dimensions.get('screen');

if (dimension.height >= dimension.width) {

return portrait\_styles

} else {

return landscape\_styles

}

}

export default ImageLogic

Висновок

При виконанні роботи я вдосконалив функціонал вкладок з книгами ( інформація про сутність та самі книги ) Тепер всі данні приходять з мережі. Так як і фільми картинки беруться з мережі та відображаються в колекції в конкретній кількості