НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №7

з дисципліни

“Програмування мобільних систем”

Виконав:

студент групи ІВ-83

ЗК ІО-7303

Ву В'єт Тунг

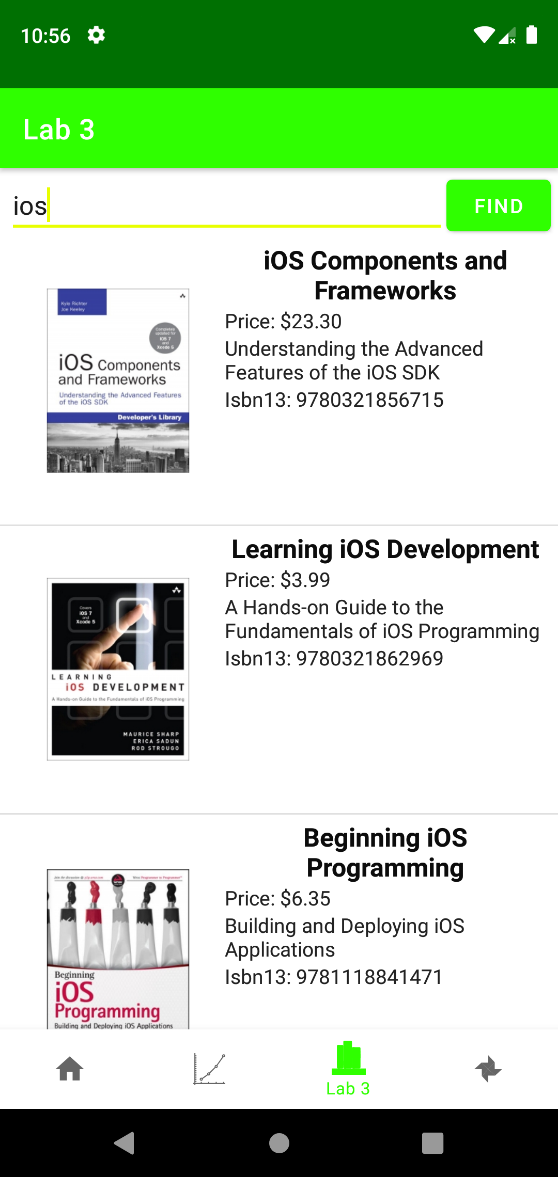
Київ 2021

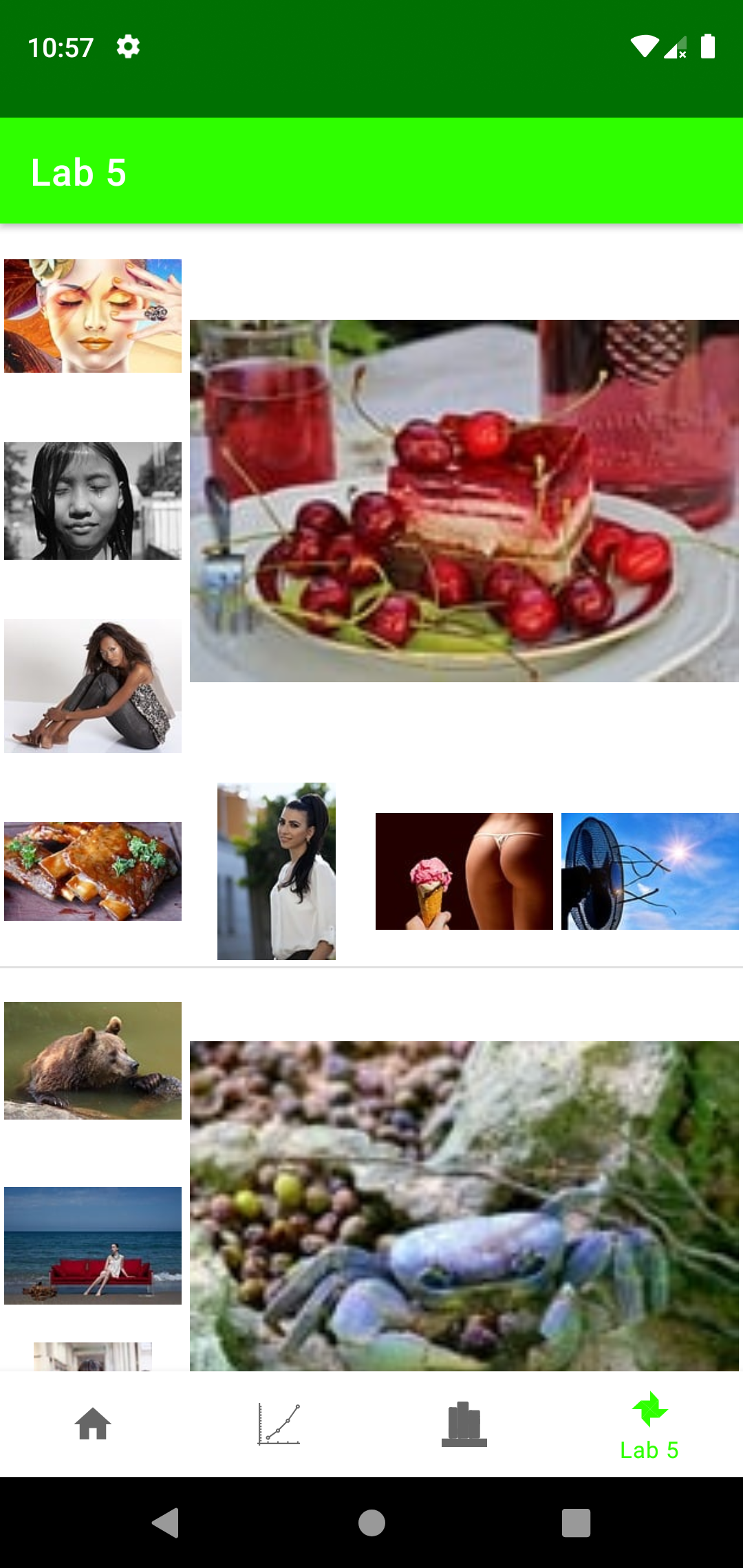
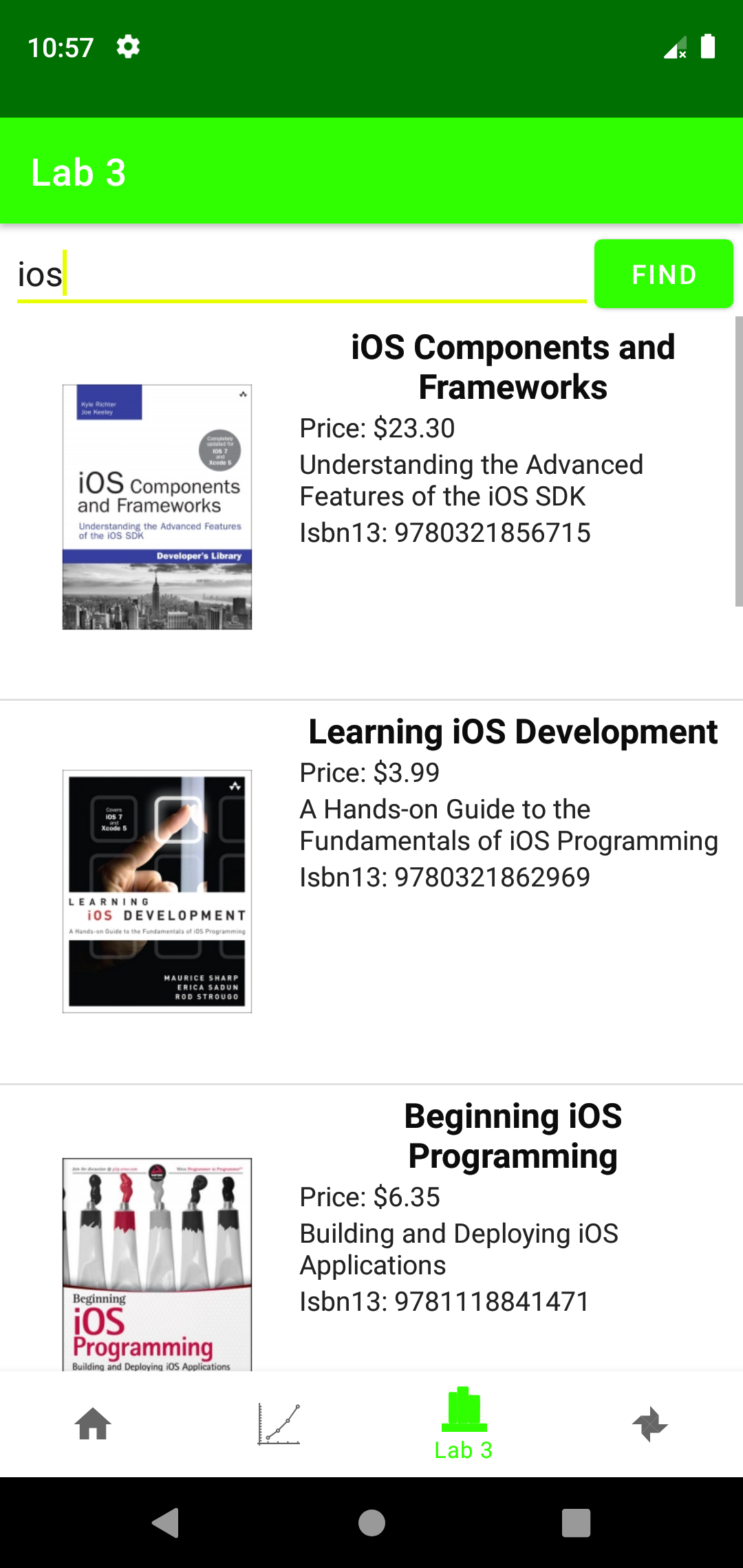
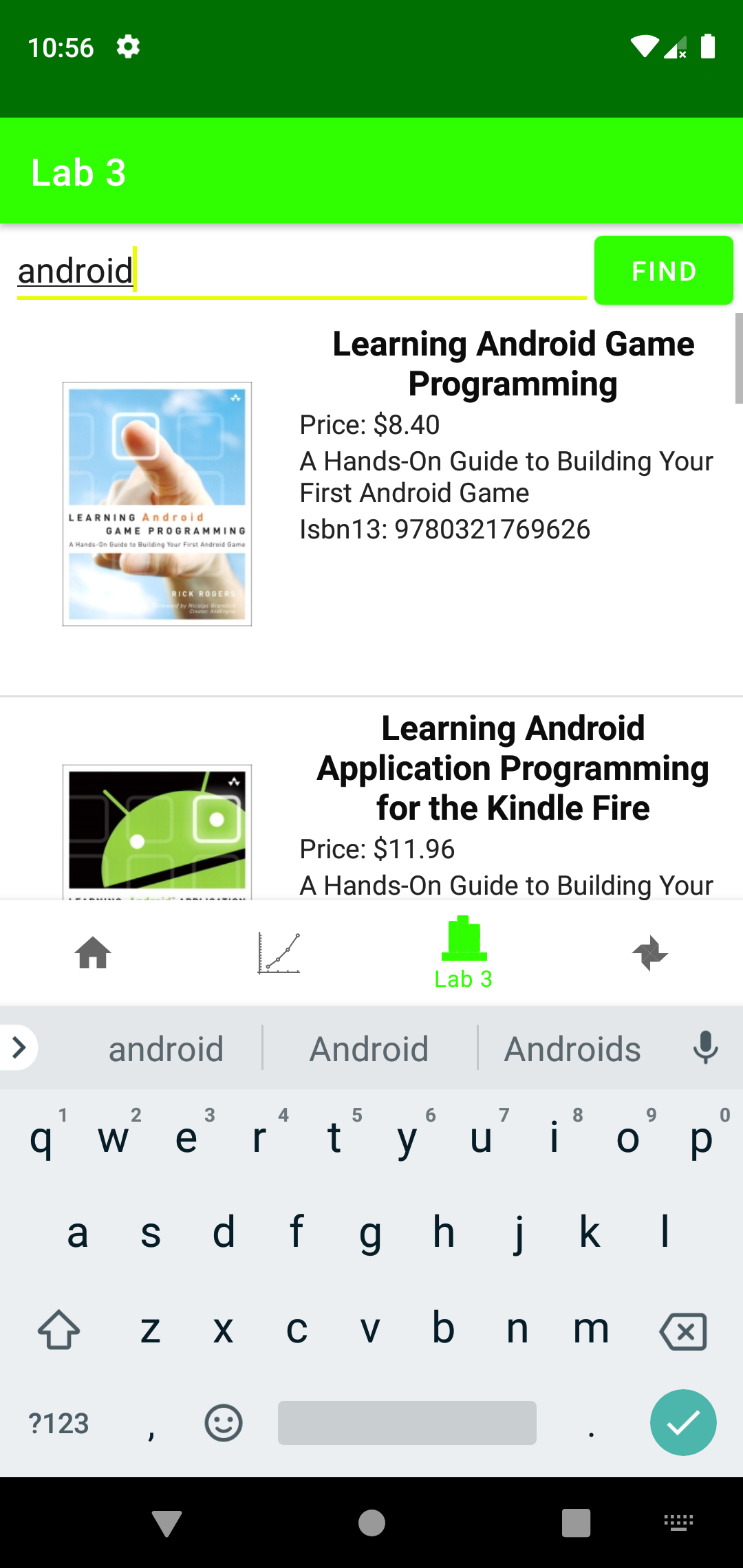
**Варіант**

**№ 7303%2+1=2**

|  |
| --- |
| **Варіант 2** |
| База даних SQLite |

**Скріншоти роботи додатка:**

****



**Лістинг коду**

**ImagesFragment.java**

package ua.kpi.comsys.IO7303.ui.images;  
  
import android.annotation.SuppressLint;  
import android.app.Activity;  
import android.content.Context;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.graphics.Point;  
import android.os.Bundle;  
import android.view.Display;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ArrayAdapter;  
import android.widget.ImageView;  
import android.widget.LinearLayout;  
import android.widget.ListView;  
import android.widget.ProgressBar;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.fragment.app.Fragment;  
  
import com.koushikdutta.async.future.FutureCallback;  
import com.koushikdutta.ion.Ion;  
  
import java.io.BufferedReader;  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.io.OutputStream;  
import java.net.MalformedURLException;  
import java.net.URL;  
import java.net.URLConnection;  
import java.security.spec.ECField;  
import java.util.ArrayList;  
import java.util.List;  
  
import ua.kpi.comsys.IO7303.R;  
import ua.kpi.comsys.IO7303.database.App;  
import ua.kpi.comsys.IO7303.database.AppDatabase;  
import ua.kpi.comsys.IO7303.database.ImageDao;  
import ua.kpi.comsys.IO7303.database.ImageEntities;  
  
public class ImagesFragment extends Fragment {  
 View root;  
 static int *width*;  
 int height;  
 private ImagesListAdapter adapter;  
 ListView listView;  
 static LinearLayout *layout*;  
 String REQUEST = "\"hot+summer\"";  
 String imageUrlTarget="\"previewURL\":\"";  
 int COUNT = 24;  
 String API\_KEY = "19193969-87191e5db266905fe8936d565";  
 View currentPage;  
 URL url;  
 static AppDatabase *appDatabase* = App.*getInstance*().getDatabase();  
 static ImageDao *imageDao* = *appDatabase*.imageDao();  
 List<List<String>> urlsLists = new ArrayList<>();  
// static AddToDB2 addToDB2 = new AddToDB2("Adder2");  
// static AddToDB2 adder = new AddToDB2();  
 Thread adderTh;  
  
  
  
 @Override  
 public void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 Display screensize = getActivity().getWindowManager().getDefaultDisplay();  
 Point size = new Point();  
 screensize.getSize(size);  
 *width* = size.x;  
 height = size.y;  
 }  
  
 @Override  
 public void onPause() {  
 super.onPause();  
 try {  
 adderTh.destroy();  
 }  
 catch (Exception e){}  
 }  
  
 public View onCreateView(@NonNull LayoutInflater inflater,  
 ViewGroup container, Bundle savedInstanceState) {  
 root = inflater.inflate(R.layout.*fragment\_four\_tab\_images*, container, false);  
 currentPage = inflater.inflate(R.layout.*images\_list*, container, false);  
  
 try {  
 url = new URL("https://pixabay.com/api/?key="+API\_KEY+"&q="+REQUEST+"&image\_type=photo&per\_page="+COUNT);  
 } catch (MalformedURLException e) {  
 e.printStackTrace();  
 }  
// Thread thread = new Thread(adder);  
// thread.start();  
  
 new ParseJson("LoadImage").start();  
  
 return root;  
 }  
  
 class ImagesListAdapter extends ArrayAdapter<List<String>> {  
 private final List<List<String>> taskImg;  
 Activity generalAct;  
  
 ImagesListAdapter(Context context, int textViewResourceId, List<List<String>> objects, Activity generalAct) {  
 super(context, textViewResourceId, objects);  
 this.taskImg = objects;  
 this.generalAct = generalAct;  
 }  
  
 @NonNull  
 @Override  
 public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {  
 LayoutInflater inflater = (LayoutInflater) getContext().getSystemService(Context.*LAYOUT\_INFLATER\_SERVICE*);  
 @SuppressLint("ViewHolder") View row = inflater.inflate(R.layout.*images\_list*, parent, false);  
  
 *layout* = row.findViewById(R.id.*imageSet*);  
 ViewGroup.LayoutParams params = *layout*.getLayoutParams();  
 params.height = *width*;  
 params.width = *width*;  
 *layout*.setLayoutParams(params);  
  
 List<ImageView> imagesListToShow = new ArrayList<>();  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img1*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img2*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img3*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img4*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img5*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img6*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img7*));  
 imagesListToShow.add(row.findViewById(R.id.*gal\_img8*));  
  
 List<ProgressBar> loadingStatusList = new ArrayList<>();  
 loadingStatusList.add(row.findViewById(R.id.*load1*));  
 loadingStatusList.add(row.findViewById(R.id.*load2*));  
 loadingStatusList.add(row.findViewById(R.id.*load3*));  
 loadingStatusList.add(row.findViewById(R.id.*load4*));  
 loadingStatusList.add(row.findViewById(R.id.*load5*));  
 loadingStatusList.add(row.findViewById(R.id.*load6*));  
 loadingStatusList.add(row.findViewById(R.id.*load7*));  
 loadingStatusList.add(row.findViewById(R.id.*load8*));  
  
 int imgNumber = taskImg.get(position).size();  
  
 for (int i=0; i<8; i++){  
 try {  
 if (i<imgNumber){  
 System.*out*.println("TASK TO SAVE IMAGE: "+i);  
 Thread.*sleep*(10);  
 LoadOrDownloadImageAndSet handler = new LoadOrDownloadImageAndSet(imagesListToShow.get(i), getActivity(), taskImg.get(position).get(i), position, getContext());  
 Thread thread = new Thread(handler);  
 thread.start();  
 }  
 else loadingStatusList.get(i).setVisibility(View.*INVISIBLE*);  
 } catch (Exception ignored){}  
 }  
  
 return row;  
 }  
 }  
  
 class ParseJson extends Thread {  
 ParseJson(String name){  
 super(name);  
 }  
  
 public void run(){  
 if (*internetAccess*()) {  
 List<String> urls = new ArrayList<>();  
 try {  
 BufferedReader br = new BufferedReader(new InputStreamReader(url.openStream()));  
 String inputLine;  
 String json = "";  
 while (true) {  
 if ((inputLine = br.readLine()) == null) break;  
 json += inputLine;  
 }  
  
 String S[] = json.split(imageUrlTarget);  
 for (String str : S) {  
 if (str.substring(0, 4).equals("http")) {  
 urls.add(str.split("\",\"")[0]);  
 }  
 }  
  
 try {  
 urlsLists.clear();  
 } catch (Exception e) {}  
  
 for (String currentUrl : urls) {  
 if (urlsLists != null) {  
 if (urlsLists.size() == 0) {  
 List<String> tempImageList = new ArrayList<>();  
 urlsLists.add(tempImageList);  
 }  
 if (urlsLists.get(urlsLists.size() - 1).size() >= 8) {  
 List<String> tempImageList = new ArrayList<>();  
 tempImageList.add(currentUrl);  
 urlsLists.add(tempImageList);  
 } else {  
 urlsLists.get(urlsLists.size() - 1).add(currentUrl);  
 }  
 }  
 }  
  
 br.close();  
  
 listView = root.findViewById(R.id.*imagesList*);  
  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 if (urlsLists != null) {  
 adapter = new ImagesListAdapter(getActivity(), R.layout.*images\_list*, urlsLists, getActivity());  
  
 listView.setAdapter(adapter);  
 } else {  
 Toast.*makeText*(getContext(), "Failed to get data", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
 });  
  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 else {  
 List<ImageEntities> imageEntities = *imageDao*.getAll();  
  
 for (ImageEntities currentEntity : imageEntities) {  
 if (urlsLists != null) {  
 if (urlsLists.size() == 0) {  
 List<String> tempImageList = new ArrayList<>();  
 urlsLists.add(tempImageList);  
 }  
 if (urlsLists.get(urlsLists.size() - 1).size() >= 8) {  
 List<String> tempImageList = new ArrayList<>();  
 tempImageList.add(currentEntity.getUrl());  
 urlsLists.add(tempImageList);  
 } else {  
 urlsLists.get(urlsLists.size() - 1).add(currentEntity.getUrl());  
 }  
 }  
 }  
  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Toast.*makeText*(getContext(), "No internet connection", Toast.*LENGTH\_LONG*).show();  
  
 if (urlsLists != null) {  
 listView = root.findViewById(R.id.*imagesList*);  
  
 adapter = new ImagesListAdapter(getActivity(), R.layout.*images\_list*, urlsLists, getActivity());  
  
 listView.setAdapter(adapter);  
 } else {  
 Toast.*makeText*(getContext(), "Failed to get data", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
 });  
 }  
 }  
 }  
  
 private static boolean internetAccess() {  
 try {  
 final URL url = new URL("http://www.google.com");  
 final URLConnection conn = url.openConnection();  
 conn.connect();  
 conn.getInputStream().close();  
 return true;  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return false;  
 }  
  
 public static class LoadOrDownloadImageAndSet implements Runnable {  
 protected ImageView imageView;  
 protected Activity uiActivity;  
 protected String imageUrl;  
 protected Context context;  
 protected int position;  
  
 public LoadOrDownloadImageAndSet(ImageView imageView, Activity uiActivity, String imageUrl, int position, Context context) {  
 this.imageView = imageView;  
 this.uiActivity = uiActivity;  
 this.imageUrl = imageUrl;  
 this.position = position;  
 this.context = context;  
 }  
  
 public void run() {  
 ImageEntities currentImage = new ImageEntities();  
 String fileName;  
  
 if (imageUrl.startsWith("http")) {  
 List<ImageEntities> daoByUrl = *imageDao*.getByUrl(imageUrl);  
 String cacheDir = context.getCacheDir() + "";  
  
 boolean imageExist = false;  
 if (daoByUrl.size() != 0) {  
 String imageCachePath = cacheDir + "/" + daoByUrl.get(0).getFileName();  
 imageExist = new File(imageCachePath).exists();  
 }  
  
 if (daoByUrl.size() == 0 | !imageExist) {  
 if (!imageExist & daoByUrl.size()>0)  
 fileName = daoByUrl.get(0).getFileName();  
 else {  
 fileName = "image\_" + hashCode() +".png";  
 while (true){  
 if (!(new File(cacheDir + "/" + fileName).exists())) break;  
 fileName = "image\_" + hashCode()+".png";  
 }  
 }  
  
 URL urlDownload;  
 try {  
 urlDownload = new URL(imageUrl);  
 InputStream input = urlDownload.openStream();  
 try {  
 OutputStream output = new FileOutputStream(cacheDir + "/" + fileName);  
 try {  
 byte[] buffer = new byte[4096];  
 int bytesRead = 0;  
 while ((bytesRead = input.read(buffer, 0, buffer.length)) >= 0) {  
 output.write(buffer, 0, bytesRead);  
 }  
 } finally {  
 output.close();  
 }  
 } finally {  
 input.close();  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 if (*imageDao*.getByUrl(imageUrl).size()==0) {  
 currentImage.url = imageUrl;  
 currentImage.fileName = fileName;  
 *imageDao*.insert(currentImage);  
// new AddToDB(imageUrl, fileName)  
// AddToDB adder = new AddToDB(imageUrl, fileName);  
// Thread thread = new Thread(adder);  
// thread.start();  
// adder.addTask(imageUrl, fileName);  
 }  
 }  
  
 try {  
 String imageNameDB = *imageDao*.getByUrl(imageUrl).get(0).getFileName();  
  
 File imageFile = new File(context.getCacheDir() + "/" + imageNameDB);  
 InputStream is = new FileInputStream(imageFile);  
  
 Bitmap userImage = BitmapFactory.*decodeStream*(is);  
  
 uiActivity.runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 imageView.setImageBitmap(userImage);  
 }  
 });  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 } else if (imageView != null) {  
 uiActivity.runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 imageView.setImageResource(R.drawable.*no\_image*);  
 }  
 });  
 }  
 }  
 }  
  
// public static class AddToDB implements Runnable {  
// protected String imageUrl;  
// protected String fileName;  
//  
// public AddToDB(String imageUrl, String fileName) {  
// this.imageUrl = imageUrl;  
// this.fileName = fileName;  
// }  
//  
// public void run() {  
// try {  
// Thread.sleep((long) Math.random()\*200+50);  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
// ImageEntities currentImage = new ImageEntities();  
// currentImage.url = imageUrl;  
// currentImage.fileName = fileName;  
// imageDao.insert(currentImage);  
//  
// }  
// }  
//  
// static class AddToDB2 implements Runnable {  
// AddToDB2(){  
//// super(name);  
// }  
// List<List<String>> tasks = new ArrayList<>();  
// List<String> task = new ArrayList<>();  
//  
//  
// public void run(){  
// while (true){  
// try {  
//// System.out.println("work.");  
// Thread.sleep(10);  
// if (tasks.size()!=0){  
// task = tasks.remove(0);  
// System.out.println(">>>>>>TASK: "+task.toString());  
// if(imageDao.getByUrl(task.get(0)).size()==0){  
// System.out.println(">>>>TSK ADD PROCESS...");  
// ImageEntities currentImage = new ImageEntities();  
// currentImage.url = task.get(0);  
// currentImage.fileName = task.get(1);  
// imageDao.insert(currentImage);  
// }  
// }  
// } catch (Exception e) {  
// e.printStackTrace();  
// }  
// }  
// }  
//  
// public void addTask(String imageUrl, String fileName){  
// List<String> temp = new ArrayList<>();  
// temp.add(imageUrl);  
// temp.add(fileName);  
// tasks.add(temp);  
// System.out.println("TASK ADDED: "+imageUrl+"; "+fileName);  
// }  
// }  
  
}

**BookDetail.java**

package ua.kpi.comsys.IO7303.ui.library;  
  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.net.MalformedURLException;  
import java.net.URL;  
  
import ua.kpi.comsys.IO7303.R;  
  
public class BookDetail extends AppCompatActivity {  
 Book book;  
 String isbn13;  
 JsonHandler jsonHandler;  
  
 @Override  
 protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*book\_info*);  
  
 Bundle arguments = getIntent().getExtras();  
 isbn13 = arguments.get("Isbn13").toString();  
  
 jsonHandler = new JsonHandler();  
 jsonHandler.*setUserFileEnable*(false);  
  
 new LoadJson("LoadBook").start();  
 }  
  
 class LoadJson extends Thread {  
 LoadJson(String name){  
 super(name);  
 }  
  
 public void run(){  
 try {  
 URL oracle = new URL("https://api.itbook.store/1.0/books/"+isbn13);  
 BufferedReader in = new BufferedReader(new InputStreamReader(oracle.openStream()));  
 String result = in.readLine();  
 System.*out*.println("RESULT: "+result);  
  
 book = jsonHandler.*importBookFromString*(result);  
  
 System.*out*.println(book);  
  
 TextView title = findViewById(R.id.*bookDetailTitle*);  
 TextView subtitle = findViewById(R.id.*subtitleDetail*);  
 TextView price = findViewById(R.id.*priceDetail*);  
 TextView rating = findViewById(R.id.*ratingDetail*);  
 TextView publisher = findViewById(R.id.*publisherDetail*);  
 TextView authors = findViewById(R.id.*authorsDetail*);  
 TextView year = findViewById(R.id.*yearDetail*);  
 TextView pages = findViewById(R.id.*pagesDetail*);  
 TextView desc = findViewById(R.id.*desc*);  
  
 ImageView imageView = (ImageView) findViewById(R.id.*imageDetail*);  
  
 runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 title.setText(book.getTitle());  
 subtitle.setText(book.getSubtitle());  
 price.setText("Price: "+ book.getPrice());  
 rating.setText("Rating: "+ book.getRating());  
 publisher.setText("Publisher: "+ book.getPublisher());  
 authors.setText("Authors: "+ book.getAuthors());  
 year.setText("Year: "+ book.getYear());  
 pages.setText("Pages: "+ book.getPages());  
 desc.setText(book.getDesc());  
  
 String imageUrl = book.getImage();  
 imageView.setImageResource(R.drawable.*no\_image*);  
  
 try {  
 new LoadAndSetImage(imageView).execute(imageUrl);  
 } catch (Exception e){  
 imageView.setImageResource(R.drawable.*no\_image*);  
 }  
 }  
 });  
 } catch (MalformedURLException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 private class LoadAndSetImage extends AsyncTask<String, Void, Bitmap> {  
 ImageView bmImage;  
  
 public LoadAndSetImage(ImageView bmImage) {  
 this.bmImage = bmImage;  
 }  
  
 protected Bitmap doInBackground(String... urls) {  
 String urldisplay = urls[0];  
 Bitmap mIcon11 = null;  
 try {  
 InputStream in = new java.net.URL(urldisplay).openStream();  
 mIcon11 = BitmapFactory.*decodeStream*(in);  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return mIcon11;  
 }  
  
 protected void onPostExecute(Bitmap result) {  
 bmImage.setImageBitmap(result);  
 }  
 }  
  
 public void setBook(Book book) {  
 this.book = book;  
 }  
}

**LibraryFragment.java**

package ua.kpi.comsys.IO7303.ui.library;  
  
import android.content.Context;  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.view.inputmethod.InputMethodManager;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.ListView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.fragment.app.Fragment;  
  
import java.io.BufferedReader;  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.io.OutputStream;  
import java.net.MalformedURLException;  
import java.net.URL;  
import java.net.URLConnection;  
import java.util.ArrayList;  
  
import java.util.List;  
import java.util.Random;  
  
import android.app.Activity;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
  
import ua.kpi.comsys.IO7303.R;  
import ua.kpi.comsys.IO7303.database.App;  
import ua.kpi.comsys.IO7303.database.AppDatabase;  
import ua.kpi.comsys.IO7303.database.BookDao;  
import ua.kpi.comsys.IO7303.database.BookEntities;  
import ua.kpi.comsys.IO7303.database.BookImageDao;  
import ua.kpi.comsys.IO7303.database.BookImageEntities;  
  
public class LibraryFragment extends Fragment {  
 private List<Book> books;  
 private List<Book> foundBooks = new ArrayList<>();  
 private List<Book> booksToShow = new ArrayList<>();  
 private BookAdapter adapter;  
 ListView listView;  
 String customBooksList = "books\_list\_custom.txt";  
 String REQUEST\_BOOK\_TITLE;  
 JsonHandler jsonHandler;  
 static AppDatabase *db* = App.*getInstance*().getDatabase();  
 BookDao bookDao = *db*.bookDao();  
 static BookImageDao *bookImageDao* = *db*.bookImageDao();  
 String searchRequest = "";  
  
  
 public View onCreateView(@NonNull LayoutInflater inflater,  
 ViewGroup container, Bundle savedInstanceState) {  
 View root = inflater.inflate(R.layout.*fragment\_third\_tab\_books*, container, false);  
  
 jsonHandler = new JsonHandler();  
 jsonHandler.*setFileUserName*(customBooksList);  
  
 books = jsonHandler.*importBookListFromJSON*(getContext());  
 listView = root.findViewById(R.id.*booksList*);  
  
 EditText searchRequest = root.findViewById(R.id.*searchField*);  
 Button searchBtn = root.findViewById(R.id.*buttonSearch*);  
  
 if(books != null){  
 adapter = new BookAdapter(getActivity(), R.layout.*book\_list*, books);  
  
 listView.setAdapter(adapter);  
 }  
 else{  
 Toast.*makeText*(getContext(), "Load failed...", Toast.*LENGTH\_LONG*).show();  
 }  
  
  
 listView.setOnItemClickListener(new AdapterView.OnItemClickListener() { //DETAIL  
 @Override  
 public void onItemClick(AdapterView<?> parent, View itemClicked, int position,  
 long id) {  
 Toast.*makeText*(getContext(), booksToShow.get((int)id).getTitle(),  
 Toast.*LENGTH\_SHORT*).show();  
  
 startActivity(new Intent(getContext(), BookDetail.class).putExtra("Isbn13", booksToShow.get((int)id).getIsbn13()));  
 }  
 });  
  
 searchBtn.setOnClickListener(new View.OnClickListener() { // FIND  
 public void onClick(View view) {  
 LibraryFragment.this.searchRequest = searchRequest.getText().toString().toLowerCase();  
 if (books!=null)  
 books.clear();  
  
 if(!LibraryFragment.this.searchRequest.equals("") & LibraryFragment.this.searchRequest.length()>=3){  
 try {  
 REQUEST\_BOOK\_TITLE = LibraryFragment.this.searchRequest;  
 REQUEST\_BOOK\_TITLE = REQUEST\_BOOK\_TITLE.replace(" ", "%20");  
  
 while (REQUEST\_BOOK\_TITLE.startsWith("%20")) REQUEST\_BOOK\_TITLE = REQUEST\_BOOK\_TITLE.substring(1);  
 while (REQUEST\_BOOK\_TITLE.endsWith("%20")) REQUEST\_BOOK\_TITLE = REQUEST\_BOOK\_TITLE.substring(0, REQUEST\_BOOK\_TITLE.length()-2);  
  
 System.*out*.println("REQUEST: "+ REQUEST\_BOOK\_TITLE);  
  
 new LoadJson("LoadJson").start();  
  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 }  
 else {  
 Toast.*makeText*(getContext(), "Uncorrected request", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
 });  
  
 return root;  
 }  
  
  
 class LoadJson extends Thread {  
 LoadJson(String name){  
 super(name);  
 }  
  
 public void run(){  
 List<BookEntities> entityByRequest = bookDao.getByRequest(REQUEST\_BOOK\_TITLE);  
 try {  
 if (!*internetAccess*() & entityByRequest.size() > 0) {  
 foundBooks = Book.*listBookEntitiesToListBooks*(entityByRequest);  
  
 jsonHandler.*exportToJSON*(getContext(), foundBooks);  
  
 BookAdapter adapter3 = new BookAdapter(getActivity(), R.layout.*book\_list*, foundBooks);  
 try {  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 listView.setAdapter(adapter3);  
 }  
 });  
 } catch (Exception e2) {  
 }  
  
 // HIDE KEYBOARD  
 InputMethodManager imm = (InputMethodManager) getActivity().getSystemService(Activity.*INPUT\_METHOD\_SERVICE*);  
 View view = getActivity().getCurrentFocus();  
 if (view == null) {  
 view = new View(getActivity());  
 }  
 imm.hideSoftInputFromWindow(view.getWindowToken(), 0);  
 }  
 else if(!*internetAccess*() & entityByRequest.size()==0){  
 foundBooks = new ArrayList<>();  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 Toast.*makeText*(getContext(), "No data in DB", Toast.*LENGTH\_LONG*).show();  
 }  
 });  
 BookAdapter adapter3 = new BookAdapter(getActivity(), R.layout.*book\_list*, foundBooks);  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 listView.setAdapter(adapter3);  
 }  
 });  
  
 }  
 else {  
 URL oracle = new URL("https://api.itbook.store/1.0/search/" + REQUEST\_BOOK\_TITLE);  
 BufferedReader in = new BufferedReader(new InputStreamReader(oracle.openStream()));  
 String result = in.readLine();  
  
 if(result==null){  
 Toast.*makeText*(getContext(), "There is no internet connection, books were not found in the database.", Toast.*LENGTH\_LONG*).show();  
 foundBooks = new ArrayList<>();  
 }  
 else  
 foundBooks = jsonHandler.*importBookListFromString*(result);  
  
 jsonHandler.*exportToJSON*(getContext(), foundBooks);  
  
 new SaveBooksToDB("save").start();  
 try {  
 if (foundBooks==null){  
 foundBooks = new ArrayList<>();  
 Toast.*makeText*(getContext(), "Nothing found", Toast.*LENGTH\_LONG*).show();  
 }  
 BookAdapter adapter3 = new BookAdapter(getActivity(), R.layout.*book\_list*, foundBooks);  
 getActivity().runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 listView.setAdapter(adapter3);  
 }  
 });  
 } catch (Exception e2) {  
 e2.printStackTrace();  
 }  
 }  
 } catch (MalformedURLException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 private class BookAdapter extends ArrayAdapter<Book>{  
 BookAdapter(Context context, int textViewResourceId, List<Book> objects) {  
 super(context, textViewResourceId, objects);  
 booksToShow = objects;  
 }  
  
 @NonNull  
 @Override  
 public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {  
 LayoutInflater inflater = getLayoutInflater();  
 View row = inflater.inflate(R.layout.*book\_list*, parent, false);  
 TextView title = row.findViewById(R.id.*bookTitle*);  
 TextView subtitle = row.findViewById(R.id.*bookSubtitle*);  
 TextView price = row.findViewById(R.id.*bookPrice*);  
 TextView isbn13 = row.findViewById(R.id.*bookIsbn13*);  
  
 title.setText(handle(booksToShow.get(position).getTitle()));  
 subtitle.setText(handle(booksToShow.get(position).getSubtitle()));  
 price.setText("Price: " + handle(booksToShow.get(position).getPrice()));  
 isbn13.setText("Isbn13: " + handle(booksToShow.get(position).getIsbn13()));  
  
 ImageView currImg = row.findViewById(R.id.*image*);  
 String imageName = booksToShow.get(position).getImage();  
 System.*out*.println(booksToShow.get(position));  
  
 try {  
 try {  
 BookImgHandler handler = new BookImgHandler(currImg, getActivity(), imageName, position, getContext());  
 Thread thread = new Thread(handler);  
 thread.start();  
 }  
 catch (Exception e){}  
  
 }  
 catch(Exception e) {currImg.setImageResource(R.drawable.*no\_image*);}  
 return row;  
 }  
  
 public String handle(String str){  
 if(str.equals("")) return "None";  
 else return str;  
  
 }  
 }  
  
 class SaveBooksToDB extends Thread {  
 SaveBooksToDB(String name){  
 super(name);  
 }  
  
 public void run(){  
 try {  
 if (bookDao.getByRequest(REQUEST\_BOOK\_TITLE).size() == 0) {  
 BookEntities bookEntity;  
 if (foundBooks != null)  
 for (Book currentBook : foundBooks) {  
 bookEntity = new BookEntities();  
 bookEntity.title = currentBook.getTitle();  
 bookEntity.subtitle = currentBook.getSubtitle();  
 bookEntity.isbn13 = currentBook.getIsbn13();  
 bookEntity.price = currentBook.getPrice();  
 bookEntity.image = currentBook.getImage();  
 bookEntity.SearchRequest = REQUEST\_BOOK\_TITLE;  
 bookDao.insert(bookEntity);  
 }  
 } else System.*out*.println("Request '" + searchRequest + "' now already in DB");  
 } catch (Exception e){}  
 }  
 }  
  
 private static boolean internetAccess() {  
 try {  
 final URL url = new URL("http://www.google.com");  
 final URLConnection conn = url.openConnection();  
 conn.connect();  
 conn.getInputStream().close();  
 return true;  
 } catch (Exception e) {  
 }  
 return false;  
 }  
  
 public static class BookImgHandler implements Runnable {  
 protected ImageView imageView;  
 protected Activity uiActivity;  
 protected String imgUrl;  
 protected Context context;  
 protected int position;  
  
 public BookImgHandler(ImageView imageView, Activity uiActivity, String imgUrl, int position, Context context) {  
 this.imageView = imageView;  
 this.uiActivity = uiActivity;  
 this.imgUrl = imgUrl;  
 this.position = position;  
 this.context = context;  
 }  
  
 public void run() {  
 BookImageEntities currentImage = new BookImageEntities();  
 System.*out*.println("Pos:"+position+"; URL:"+ imgUrl);  
 String fileName;  
  
 if (imgUrl ==null)  
 imgUrl ="";  
 if (imgUrl.startsWith("http") | imgUrl.startsWith("https")) {  
 System.*out*.println(">>>>>>>>>>>>>>>>>ENTER BLOCK");  
 List<BookImageEntities> daoByUrl = *bookImageDao*.getByUrl(imgUrl);  
 String cacheDir = context.getCacheDir() + "";  
  
 boolean imageExist = false;  
 if (daoByUrl.size() != 0) {  
 String imageCachePath = cacheDir + "/" + daoByUrl.get(0).getFileName();  
 imageExist = new File(imageCachePath).exists();  
 System.*out*.println("FILE:"+daoByUrl.get(0).getFileName()+"; Exist:"+imageExist);  
 }  
  
 if (daoByUrl.size() == 0 | !imageExist) {  
 if (!imageExist & daoByUrl.size()>0)  
 fileName = daoByUrl.get(0).getFileName();  
 else {  
 int rndInt = new Random().nextInt(9999);  
 fileName = "image\_" + hashCode()+ "\_" + rndInt+".png";  
 }  
  
 URL urlDownload;  
 try {  
 urlDownload = new URL(imgUrl);  
 InputStream input = urlDownload.openStream();  
 try {  
 OutputStream output = new FileOutputStream(cacheDir + "/" + fileName);  
 try {  
 byte[] buffer = new byte[2048];  
 int bytesRead = 0;  
 while ((bytesRead = input.read(buffer, 0, buffer.length)) >= 0) {  
 output.write(buffer, 0, bytesRead);  
 }  
 } finally {  
 output.close();  
 }  
 } finally {  
 input.close();  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 System.*out*.println(">>>>>>>>>>>>URL IMAGES BEFORE ADDING: "+ imgUrl);  
 currentImage.url = imgUrl;  
 currentImage.fileName = fileName;  
 *bookImageDao*.insert(currentImage);  
 }  
  
 try {  
 System.*out*.println("TRY SET CACHED IMAGE");  
 String imageNameDB = *bookImageDao*.getByUrl(imgUrl).get(0).getFileName();  
  
 File imageFile = new File(context.getCacheDir() + "/" + imageNameDB);  
 InputStream is = new FileInputStream(imageFile);  
  
 Bitmap userImage = BitmapFactory.*decodeStream*(is);  
  
 uiActivity.runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 imageView.setImageBitmap(userImage);  
 }  
 });  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
  
 } else if (imageView != null) {  
 uiActivity.runOnUiThread(new Runnable() {  
 @Override  
 public void run() {  
 imageView.setImageResource(R.drawable.*no\_image*);  
 }  
 });  
 }  
 }  
 }  
}

**Висновок**

Виконано cьому лабораторну роботу. Модифіковано третю та четверту вкладки, додаток завантажує дані з мережі та зберігає їх у БД SQLite.