MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
VOLODYMYR DAHL EAST UKRAINIAN NATIONAL UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY AND ELECTRONICS
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Admitted to defending
Head of the CSE department

_____ I. S. Skarha-Bandurova
«____»_____ 2020

UNDERGRADUATE PROJECT (WORK)
EXPLANATORY NOTES
ON THE TOPIC:

Android application "Handbook for Celebration"

Bachelor's degree

Speciality 122 Computer Science
(code and name of the specialty)

Supervisor: _____ (signature) E. V. Shcherbakov (initials, surname)

Occupational safety consultant: _____ (signature) Ya. O. Krytska (initials, surname)

Applicant for higher education: _____ (signature) M. O. Babayeva (initials, surname)

Group: KH-16d

Severodonetsk 2020

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
VOLODYMYR DAHL EAST UKRAINIAN NATIONAL UNIVERSITY

Faculty Information technology and electronics
Department Computer Science and Engineering
Educational degree Bachelor
Specialty 122 Computer Science
(code and name)

APPROVED:
a.i. head of the CSE department
_____ S.O. Safonova
___
«_____» _____2020

TASK
FOR THE UNDERGRADUATE PROJECT (WORK)

Maya Orazmuradovna. Babayeva
(Full Name)

1.Theme of work Android application "Handbook for Celebration"

Project manager (works) Assoc. Prof., Dr. Shcherbakov E.V.

Approved by order of the higher educational institution from "30" 04 2020 № 78/15.15
2.Deadline for applicants for higher education 08.06.2020
3. Initial data to work Used software: Android Studio for mobile application development in Java and XML

4. Contents of the settlement and explanatory note (list of questions required to develop):
1) Analysis of the subject area
2) Selection and justification of software for the system development
3) System development
4) Labor protection
5) Conclusions

5. List of graphic material (with exact indication of obligatory drawings)
Electronic posters

6. Consultants of project section (works)

Section	Surname, initials and position of the consultant	Signature, date	
		Task issued	Task accepted
Occupational Health	Y.O. Krytska, Senior lecturer		

7. Date of issuance of the task _____

Supervisor _____
(signature)

Applicant for higher education _____
(signature)

SCHEDULE PLAN

№ s/n	The name of the stages of the diploma project (work)	Term of execution of project stages (works)	Note
1	Inspection of literature behind the topic of the diploma	5.05.20-09.05.20	
2	Study of literary sources	11.05.20-15.05.20	
3	Development of the technical task	16.05.20-19.06.20	
4	Designing to design	21.05.20-22.05.20	
5	Program Structure	23.05.20-29.05.20	
6	Development of the mobile app	31.05.20-05.06.20	
7	Execution of an explanatory note	06.06.20-16.06.20	

Applicant for higher education _____ M. O. Babayeva
(signature) (initials, surname)

Supervisor _____ E. V. Shcherbakov

ABSTRACT

Explanatory note to the diploma project (work) of the bachelor degree in Computer Science of V. Dahl East Ukrainian National University: 90 pages, 10 figures, 4 tables, 22 references.

Object of development: Handbook for Celebration

Goal: design and development Android application "Handbook for Celebration".

The project is completed:

1. Project development with Android studio using Java and XML.
2. Construction and Implementation of the application.
3. A mobile application for choosing the best gift for any occasion has been developed.

Practical value, scope of work: the mobile application provides users with menu of functions, such as gift selection, holiday decoration advice, as well as sections with horoscope information.

THE MOBILE APPLICATION, ANDROID, SERVICE, HANDBOOK FOR CELEBRATION, MARK-UP XML AND JAVA LANGUAGES

Conditions for obtaining a BSc project: V. Dahl EUNU, 59-a Central Avenue, Severodonetsk, 93400.

Content

INTRODUCTION

Today, the development of a mobile application is one of the popular topics all over the world that are being actively developed [1]. After the advent of smartphones according to statistics, a significant number of people use it during the day for business and leasure purposes.

Creating a mobile application provides new opportunities for expanding, information support or advertising of a business.

A professionally created application gives the user the ability to query the search engines depending on what purpose it was created. Constantly downloading, the application enables programmers to develop experience in the IT field without advertising.

A quality application is a basic information reference for users. Using a mobile application, you can [2]:

- view information on the topic at any convenient time;
- international online payment;
- online shopping for goods and services;
- electronic dictionaries;
- WPS Office editing;
- banking application and others.

The goal of the graduation project is to develop a mobile application for OS Android "Handbook for Celebration" intended for any user containing information about different gifts for any type of holiday, tips and information on the horoscope, etc.

1 ANALYSIS OF MEANS OF DEVELOPING APPLICATIONS FOR ANDROID OS

1.1 Domain Overview

Android is an operating system for mobile phones based on the Linux kernel [3]. This system is quite recent realted to others; the first device HTC Dream (T - Mobile G1) with OS Android was introduced in September 2008. For comparison: the first device with WinMobile was April 2000, Symbian Ltd was founded in 1998, and the first smartphones with this system on board also appeared in 2000. Let's get back to Android: first, this system was developed by Android Inc., which was then bought by Google.

In this year Google initiated the creation of the Open Handset Alliance (OHA) business alliance, which is now engaged in support and further development of platforms. OHA currently has 65 companies, including mobile phone manufacturers, software developers, some mobile providers, and chip manufacturers. Among them: Google, HTC, Intel, Motorola, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile, Nvidia. Android platform is already very popular in the market of mobile devices.

So, according to research of NPD Group inc. [4], already in the first quarter of 2010 Android OS outperformed the iPhone OS in popularity. According to NPD, today Google has 28% of the market, which puts them in second place after Research in Motion (36%), iPhone OS is in third place with 21%.

Also at Mobile World Congress 2010 was announced several tablets and netbooks with Android as the operating system, created work, the control device of which smartphones with Android OS. Also on Google IO in 2010, Google TV was announced, also based on Android OS.

1.2 Java programming language

First, a Java speech called Oak ("Oak") was developed by James Gosling for programming consumer electronic devices [5]. Subsequently, it was renamed Java and was used to write client applications and server software. It is named after the Java coffee brand, which in turn, received the name of the island of the same name (Java), therefore a cup with hot coffee, and is depicted on the official language emblem. There is another version of the origin of in the name of the language associated with an allusion to a coffee machine as an example of a household device for the programming of which speech was first created.

Java is an object-oriented programming language developed by Sun Microsystems (later acquired by Oracle). Java programs can run on any virtual Java machine, regardless of computer architecture. The official release date is May 23, 1995.

The Java language is actively used to create mobile applications for the Android operating system. Application development can be done in Android Studio, NetBeans, Eclipse, using the Android Development Tools (ADT) plugin or in IntelliJ IDEA. The JDK version should be 5.0 or higher.

Java was chosen as the programming language for the application , since Java has a large selection of libraries and is officially supported in Android Studio , which was chosen as the application development environment .

Android Studio has replaced the flags on ADT for the Eclipse platform. The environment is based on the source code of the product IntelliJ IDEA Community Edition by the developing company JetBrains. Android Studio is developed as part of an open development model and is distributed under the Apache 2.0 license.

Binary assemblies are prepared for Linux (for testing and the state of Ubuntu), Mac OS X and Windows. Media still provides the means for the development of IP Nij not only for smartphones and tablet, but also for handheld devices based on of Android Wear app, TV (All Android the TV), the eye piece and the Google Glass, and automotive infotainment systems (All Android the Auto). For applications,

originally designed with IC Niemi Eclipse and the ADT the Plugin, prepared by the tool to automatically import existing project in All Android Studio.

The development environment is adapted to perform typical tasks that are solved in the process of developing applications for the Android platform. Including environment includes tools to simplify test programs for compatibility with different versions of the platform and tools for designing applications, running on devices with screens of different resolutions (tablets, smartphones, laptops, watches, glasses, etc.). In addition to the features that are present in IntelliJ IDEA, Android Studio implements several additional functions, such as a new unified subsystem for compiling, testing and deploying applications, based on Gradle assembly tools and supporting the use of continuous integration tools.

To speed up application development, a collection of typical interface elements and a visual editor for compiling them are presented, it provides a convenient preview of the various states of the application interface (for example, you can see how the interface will look for different versions of Android and for different screen sizes). To create custom interfaces, there is a wizard to create your own design elements, supports the use of templates. The environment has built-in functions for loading typical code samples from GitHub .

The structure also includes advanced refactoring tools adapted to the features of the Android platform, checking compatibility with previous releases, identifying performance problems, monitoring memory consumption and evaluating usability. A quick edit mode has been added to the editor. The system of highlighting, static analysis and error detection is expanded with support for the Android API. Integrated ProGuard code optimizer support. An interface for managing translations into other languages is provided.

1.3 Android Studio

Android Studio [6] has the following features:

1. Live layouts: live coding, view (rendering) in real-time program.

2. Developer Console: optimization tips, translation assistance, direction tracking, campaigning and promotions, Google analytics metrics.

3. Based on Gradle.

4. Android refactoring and quick fixes.

5. Lint utilities to cover performance, usability, version compatibility, and other issues.

6. Using ProGuard features and program signatures.

7. Templates for creating common Android designs and components.

8, A rich editor of layouts (layouts) that allows users to drag and drop the components of the user interface, as an option, to see simultaneously layouts on various screen configurations.

As the development environment of the Android program, the software product Android Studio was selected. It has an integrated development environment (IDE) for working with the Android platform, announced May 16, 2013 at the Google I/O conference .

1.4 XML markup language

The XML language [7] is designed to store and transfer data. In the same time the HTML language is designed to display data.

Before proceeding, make sure you have basic knowledge of HTML. If you don't know what HTML is, then the HTML tutorial for beginners will help you figure this out.

XML stands for English as eXtensible Markup Language. XML is a markup language that resembles HTML. XML is for data transfer, not for displaying it. XML

tags are not predefined. You must determine the tags you need. XML is described in such a way as to be self- defined .

The differences between the XML and HTML languages are as follows.

XML is not a replacement for HTML. They are designed to solve various problems: XML solves the problem of storing and transporting data, focusing on what this very data is, HTML solves the problem of displaying data, focusing on how this data looks. Thus, HTML takes care of the display of information, and XML takes care of the transport of information.

Perhaps you will find it a little strange, but XML does not do anything. It was created to structure, store and transmit information.

The following example presents a note from Tovi to Janie, saved in XML format:

```
<? xml version = "1.0" encoding = "UTF-8"?>
< note >
  < to > Tove </to>
  < from> Jani </from>
  < heading > Reminder </ heading >
  < body > Don't forget about me this weekend! </ / body >
</ note >
```

The above record is quite descriptive. There is information about the sender and receiver. There is also header data and the message itself. And for all that, this XML document does nothing. This is just information wrapped in tags. Someone must write a program that will send, receive, and display this data.

Tags in the above example (for example, <to> and <from>) are not defined by any XML standards. These tags were "invented" by the author of this XML document. That's because there are no predefined tags in the XML language.

So, in HTML, all tags used are predefined. HTML documents can only use tags that are defined in HTML standards (<p>, < li >, etc.).

XML allows the author to define his language tags and his document structure. XML is not a replacement for HTML XML is a complement to HTML.

It is important to understand that XML is not a replacement for HTML. Most web applications use XML to transport data, and HTML to format and display data.

Nowadays, XML is also important for the web, as HTML was once important for the birth of the modern Internet. XML is a common tool for transferring data between all kinds of applications.

XML is used in many aspects of web development, but its main task is to facilitate the storage and transfer of data.

If you need to display dynamic data in an HTML document, it will take too much time, if whenever this data has changed, edit the HTML document itself.

With XML, data can be stored in separate XML files . At the same time, you focus on using HTML/CSS to render and template and you can be sure that new data arriving will not require any changes to the HTML code of the document.

In the real world, computer systems and databases use data in incompatible formats.

XML data is stored in plain text format. This provides software and hardware independence. This makes it easy to create data that can be used by a wide variety of applications.

One of the most time-consuming problems of developers has always been and still is the problem of data exchange between incompatible systems.

Transferring data in the form of XML significantly reduces the complexity of this problem, since data in this format can be read by various incompatible applications.

The transition to new systems (hardware or software platforms) always takes a lot of time. A lot of data needs to be converted to new formats. However, often incompatible data is lost.

XML data is stored in text format. This greatly facilitates the expansion or modernization of operating systems, the transition to new applications or browsers without the risk of losing data.

Access to your data can get not only HTML documents, but also any other applications. Thanks to XML, your data becomes available for all types of "reading machines" (voice machines, news channels, etc.), which makes it much easier for people with visual impairments and other physical problems to access them.

Using XML, many programming languages have been created on the Internet. Here are some examples:

- XHTML;
- WSDL to describe available web services;
- WAP and WML as markup languages for handheld devices such as PDAs;
- RSS languages for news feeds;
- RDF and OWL for resource description and ontology;
- SMIL to describe multimedia for the network.

The layout defines the visual structure of the user interface, for example, the operation user interface or application widget. There are two ways to declare a layout:

Declaring user interface elements in XML. Android has a handy reference for XML elements for View classes and their subclasses, such as those used for widgets and layouts.

Creating instances of elements at runtime. Your application can programmatically create View and View Group objects (as well as manage their properties).

The Android platform gives you the flexibility to use any of these methods to declare and control the user interface of an application. For example, you can declare default layouts in XML, including screen elements that will be displayed in layouts and their properties. Then you can add code to the application that allows you to change the state of objects on the screen (including those declared in XML) at run time.

The ADT plug-in for Eclipse(what is this stands for ?) provides a preview function for the XML file you created - just open the XML file and select the Layout tab .

You can use the Hierarchy Viewer tool to debug layouts - you can use it to view property values, frames with fill or field indicators, and fully rendered views right while debugging the application in the emulator or on the device.

Using the layout opt tool, you can quickly analyze layouts and their hierarchies for poor performance or other problems.

The advantage of declaring a user interface in an XML file is that in this way you can more effectively separate the presentation of your application from the code that controls its behavior. User interface descriptions are outside the scope of your application code. This means that you can change or adapt the interface without having to make changes to the source code and recompile it. For example, you can create different layout XML files for screens of different sizes and different screen orientations, as well as for different languages. Additionally, declaring a layout in XML simplifies the visualization of the user interface structure, making debugging problems also easier. In this article, we will teach you how to declare a layout in XML. If you prefer to instantiate View objects at run time, see the reference documentation for the View Group and View classes .

Typically, a reference to XML elements for declaring user interface elements exactly follows the structure and naming rules for classes and methods - element names correspond to class names, and attribute names correspond to methods. In fact, the correspondence is often so exact that you can easily guess which XML attribute corresponds to a particular class method, or which class corresponds to a given XML element. However, it should be noted that not all directories are identical. In some cases, the names may vary slightly. For example, the element Edit Text has an attribute text, which corresponds to the method EditText.setText().

1.5 Application Components

As it was mentioned above, Android is an operating system based on Linux with a programming interface on the Java [2]. This system is equipped with tools such as the compiler, debugger and device emulator, as well as its (Android) Java

virtual machine (Dalvik Virtual Machine - DVM). Android was created by the Open Handset Alliance, led by Google.

Android uses a special virtual machine called the Dalvik Virtual Machine. Dalvik uses its own special byte code. Therefore, you cannot run the standard Java bytecode on Android. Android provides a dx tool that allows you to convert Java Class files to .dex files (Dalvik Executable). Android applications are packaged in .apk files (Android Package) with the aapt (Android Asset Packaging Tool) program. To simplify development, Google provides Android Development Tools (ADT) for Eclipse. ADT automatically converts from Java Class files to .dex files, and creates .apk during deployment.

1.5.1 The main components of Android

Android applications consist of the following parts:

Activity - is a scheme for representing Android applications. For example, it could be the screen that the user sees. An Android application can have several activations and can switch between them at runtime.

Views - the activity user interface created by widgets of classes inherited from android.view.View class. The views schema is managed through android.view.ViewGroups class.

Services - performs background tasks without providing a user interface. They can notify the user through the Android notification system.

Content Provider - provides data to applications, with the help of a content provider your application can exchange data with other applications. Android contains a SQLite database , which may be a content provider.

Intents are a synchronous messages that allow an application to request functions from other services or activations. An application can do direct intentions to a service or activity (explicit intention) or request from Android registered services and applications for intent (implicit intention). For example, an application can request via intent a contact from the contacts application (phone / address book) of

the device. The application registers itself in intents through Intent Filter . Intents is a powerful concept that allows you to create loosely coupled applications.

Broadcast Receiver/Broadcast Receiver (hereinafter referred to simply as the Receiver) - receives system messages and implicit intents, can be used to respond to system status changes. An application can register as a receiver of certain events and can be launched if such an event occurs.

Other parts of Android are widgets, or live folders (Live Folders), or live wallpapers (Live Wallpapers). Live folders display the source of any data on the desktop without launching the corresponding applications.

1.5.2 AndroidManifest.xml file

Android applications are described by the AndroidManifest.xml file. All activity, services, receivers and content providers of the application must be declared in these files. It should also contain the permissions required by the application.

1.5.3 R.java, Resources and Assets

The gen directory in an Android project contains the generated values.

While the res directory stores structured values known to the Android platform, the assets directory can be used to store any data. In Java, you can access this data through the Assets Manager and the getAssets() method.

1.5.4 Activity and layout

The user interface for the activity is determined with the help of layouts. At runtime, layouts are instances of android.view.ViewGroups class. The layout defines the user interface elements, their properties and location. UI elements are based on the android.view.View class. The layout can be defined using Java code or using XML.

1.5.5 Activity and life cycle

The operating system controls the life cycle of your application. The most important methods:

onSaveInstanceState() - calls if activity is stopped. It is used to save data when restoring the activity state , if the activity is resumed;

onPause() - always called if activity has completed, can be used to free resources or save data;

onResume() - called if activity is resumed, can be used to initialize fields.

1.5.6 Context

The android.content.Context class represents connections to the Android system. This is the interface for global information about the application environment. The context also provides the getSystemService() method, which allows you to get the control object for various pieces of equipment. Since Activities and Services extend the Context class, you can directly access the context with "this".

1.6 Resources application

This section describes how to create, package, and use the resources of your application with strings, images, and files. For example, you can pack a file along with a casual game containing definitions of game levels and load it at run time. We will also consider how ensuring independence of resources from the application logic makes it easy to localize and configure the application for different countries, displays, accessibility settings and other contexts with different users and computers. Resources, such as strings and images, should usually be presented in several versions for different languages, scales and contrast. For such resources, support is provided for a resource management system.

There are two types of application resources.

A file resource is a resource stored as a file on disk. A file resource can contain bitmap, XAML, XML, HTML, or any other data.

Nested resource - A resource embedded in a specific file containing resources. The most common example is a string resource located in a resource file (RESW or RESJSON file).

1.7. Analysis of existing analogues

The Holiday Calendar application [9] was taken as the main analogue and prototype (see fig. 1.1). I analyzed several applications to identify differences between them and make requirements for my own implementation.

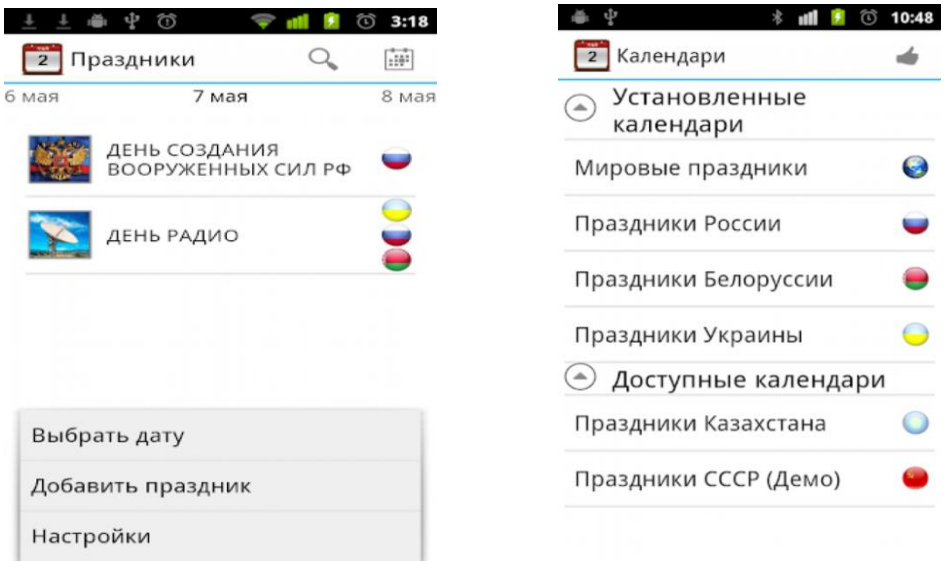

Figure 1.1 - Example of the main menu of the "Holiday Calendar" application

As can be seen from fig.1.1, this Holiday Calendar provides information about which holidays are celebrated on a particular day of any year. The calendar contains a description of national holidays in Ukraine, Russia, and Belarus, as well as a large number of international holidays and observances. Dates of rolling holidays are recalculated automatically for each year. You can easily find the holiday you are

interested in or find out what you can congratulate your friends tomorrow, the day after tomorrow, or maybe in a couple of years. The program does not require an Internet connection. An example of displaying the application "Holiday Calendar" is shown in fig. 1.2.

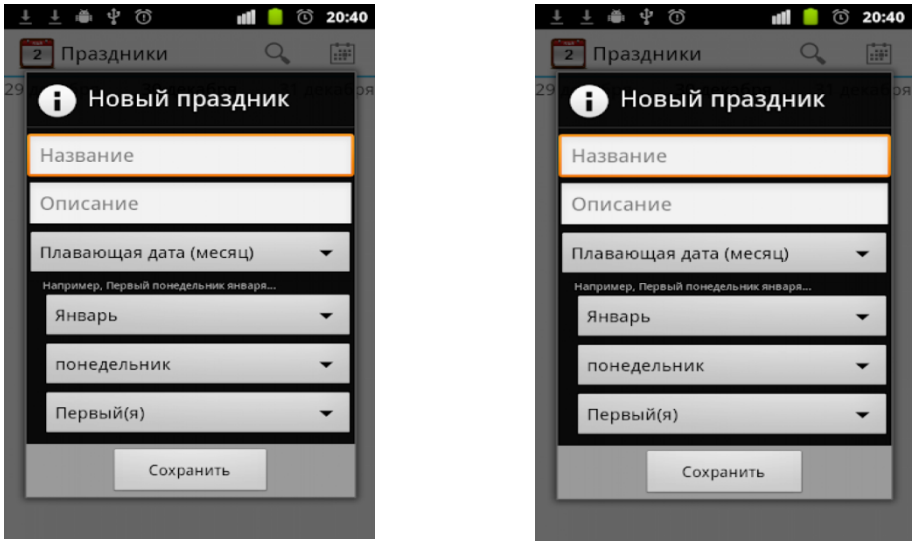

Figure 1.2 - An example of displaying the application "Holiday Calendar"

After analyzing all the advantages and disadvantages, a technical task was drawn up to develop an application on the Android OS using XML, Java.

1.7 Requirements Specification for a Mobile Application Development

Purpose: A Handbook for Celebration is the smartphone app that helps people to organize their special events like dates, birthdays, etc. This application should run on a mobile platphorm.

Intended audience: people from 14 to 100 who use the smartphones

Operating environment and target OS platform: Android

Product features: Android application "Handbook for Celebration" should contain for any user that contains information about the different gifts for every type of holiday, advice and information on a horoscope.

The application should include the following sections:

1. Flowers
2. Jewelry
3. Toys
4. For parents
5. Information
6. Advice

Other features: view parameters for various gifts, tips or horoscope information for various types of holidays in one mobile application.

The application should provide the user with the easiest way to view options for various gifts, tips or horoscope information. The user can be informed about different gifts for any type of holiday, event or celebration.

To achieve these goals, the application must solve the following tasks:

- providing information on the category ;
- providing information on the horoscope ;
- providing a short description about the application.

1.8 Conclusions to the first section

The purpose of the graduation project is to develop a Android application "Handbook for Celebration" using the Android Studio software. The application should implement the choice of the best gift for any type of holiday by category, horoscope, advice, information about the application and a convenient interface.

2 DESIGNING APPLICATION USING XML, JAVA

2.1 Analysis and use of XML

XML is an eXtensible Markup Language. Extensibility of this language allows configure everything according to needs; create own tag sets for any purpose [7, 10]. It's called a markup language because it uses "tags" to determine what needs to be done so. In most cases, this language is similar to the HTML5 language which is used for website development and application design nowadays.

As opposed to programming languages, markup languages use tags and attributes in these tags, as well as nesting to accomplish tasks. XML is used for design, the user interface, user experience, styles and themes, graphics, and etc. Moreover, learning XML markup is much easier than learning software structures and language concepts.

XML markup is contained in simple text files with the extension .xml. XML files can be created in a text editor, for example, in Notepad; however, most programmers use markup language design tools such as Eclipse, IntelliJ, or NetBeans. Later created XML files are opened or parsed by the Android OS or Java application code and converted into Java subject pattern using XML "detections" in each file.

2.2 Analysis and use of Java

There are three main versions of the Java language: SE (Standard Edition) for individual users, EE (Enterprise Edition) for large user groups and ME (Micro Edition) for legacy mobile flip phones [11]. Most modern Android smartphones use Java SE, not Java ME [8].

What is interesting for the Android OS, the standard version of Java (Java SE 6 or 7) is used, just like it does on a PC.

2.3 XML naming schemes

XML consists of tags and their attributes [10]. Attributes are part of tags and are used to configure tag functions, as well as for links of new media assets, fonts, color values, styles, themes, other XML definitions and similar assets that may be required to format or define how an application element is displayed on the user's screen.

XML tags and attributes that can be used with a specific framework, like in Android, use the XML naming scheme. Such a scheme for Android is contained in a centralized Web Repository of Google's Android servers.

As XML was originally designed "extensible" it should have naming schemes, which means that there is no "standard" version of XML; Each version is configured according to requirement of the end-user. In the case of XML for Android, it was specifically created for Android application development.

For example, Android developers created an <RelativeLayout> XML tag for UI design based on a layout with relative placement of elements, which ultimately uses the RelativeLayout Java class to define objects of this class. The following is a listing of the activity_main.xml file for a simple UI application using this layout:

```
<RelativeLayout
        xmlns:android=
"http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight=
"@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
```

```
    tools:context=".MainActivity" >
    <TextView
style="@style/WelcomeTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/welcome" />
    <ImageView
 ...
/>
                                ...
</RelativeLayout>
```

To check all tags and attributes are valid and "valid" the XML naming scheme is referenced at the very top of each XML detection file, so the XML markup inside of this file can be checked according its XML naming scheme (configuration specification). Eclipse - ADT does not do it in "real time," so an active Internet connection is not required to develop XML markup. The process of validating tags and attributes (according to the definitions of the XML naming scheme) is called XML validation.

In any eXtensible markup language, the URL of the naming scheme must be in the first (external) "parent" tag. This parent tag typically contains "child" tags that are "nested" inside of the parent tag. For more clarity, nesting of tags is usually marked with indents relative to each other.

Android OS actually has two XML naming schemes nested in two different repositories. The first scheme, the XML naming scheme for Android packages (apk), was designed for high-level design-oriented XML and is located at schemas.android.com/apk. This scheme is used in 95% of the application development process. The second scheme is the naming scheme for the Android toolkit (tools) designed for low-level XML, oriented to OS usage, and is located in

schemas.android.com/tools. This XML can be used, for example, to declare objects of the Context class.

2.4 Java Name Scopes

As a language that supports dynamic method of loading by the Internet Java prevents name conflicts [8]. Therefore, in particular, in Java there are no global variables, all variables and functions (methods) are parts of the class, moreover, each class is part of the package.

In Java each variable or method is referenced by using a full name consisting of the package name, class name, and class member name, separated by dots. A package name usually consists of many components, also separated by dots. Java developers propose using the domain names of package development organizations in package names to ensure the uniqueness of package names on the Internet, for example: en.bmstu.students.rk6.Ivanov.game

Class files are stored in folders according to the package name. For example, all game classes of student Ivanov are stored in a folder with a relative name: com / bmstu / students / rk6 / Ivanov / games

To determine the *full* path name of the folder containing the package class files, the Java interpreter java uses the CLASSPATH system variable, which can be set, for example, with bash the command:

export CLASSPATH = .: ~ / classes: / usr / local / java / classes

It is not necessary to include paths to system classes in CLASSPATH, since the Java interpreter attaches these paths to the list on the right by itself.

Classpath parameter of the java interpreter plays a similar but not quite similar role.

To include a class in a package, use the instruction:

package <package_name>;

which must be the first instruction in the source file. If the package statement is not in the source code of the class, then the class is considered to belong to the "default" package, whose class files are located in the current folder, which is convenient for development.

An optional package statement can be followed by any number of import statements that have two forms of notation:

import <package-name>. <class-name>;
import <package_name>. *;

The purpose of the import statement is to enable the use of "shortened" class names and their members in the program text (without the prefix in the form of a package name). It is important to understand that the import statement does not entail any loading of individual classes or packages (this is its difference from #include in C ++). Once again - classes in Java are loaded dynamically only when they are needed ! Access to packages, classes, and class members is controlled by using the public, protected, and private modifier keywords and is governed by the following rules.

The package is available if the appropriate files and folders are available (for example, if local files have appropriate read permissions or can be downloaded over the network).

Each class and interface of a package is accessible to all other classes and interfaces of the same package. It is impossible to define Java classes that are visible in only one source file.

A class designated as public in one package is available from another package if the package itself is available. Classes not designated as public are not available outside of the package.

Members of a class are accessible from another class in the same package unless they are designated private. Members of a class designated as private are only available in their own class.

A member of class A is accessible from class B, which is part of another package, in cases when class A, like its member, is designated as public or when class A is designated as public, class member is designated as protected, and class B is derived class of class A.

All members of the class are always available within the class.

2.5 . Tag nesting

Any tag that is used as a parent tag must have a closing tag with a slash as a prefix before the tag name: <RelativeLayout> ... </RelativeLayout>.

Alternatively, a child tag that does not have its own child tags should have only a closing slash at the end of the opening tag: <TextView android : attributes />.

Tags can be nested into each other to a depth greater than one, for example:

```
<RelativeLayout>
        <LinearLayout>
                <TextView />
                <ImageView />
        </LinearLayout>
</RelativeLayout>
```

2.6 . Links to XML files. File Strings.xml

XML files can refer to other XML files, so a chain can be created to make XML definitions more modular. Links between XML files are somewhat similar to nesting tags, but extend through files. Links in attribute values begin with the @ symbol, which is specific to links between files in Android.

For example, the prefixes @ string /, @ style / and @ dimen / used in the attribute values of the activity_main.xml file given above refer to the tags and attributes of the strings.xml, styles.xml and dimensions.xml files, respectively.

The following is an example of a listing of the strings.xml file located in the / res / values folder of a project:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
        <string name="app_name">Welcome</string>
        <string name="welcome">Привет, Андроид!</string>
...
</resources>
```

The strings.xml file does not have a reference to the XML naming scheme in the Android repository, since it contains only resource definitions. It contains the parent <resources> tag and child <string> tags, each of which consists of a variable name in the name attribute (for example, name = "welcome") and its value (Hello, Android!) between the start and end tags.

An example of using a style rule in a control located in the activity_main.xml activity file:

```
 <TextView
        ...
        android:text="@string/welcome"
        ...
```

2.7 Definition and application of styles: file styles.xml

Styling UI elements in Android applications is as follows. First, in the resource file (styles.xml file in the res / values folder of the application), style rules are defined

containing the attribute_name / value pairs for the group of GUI components. Then these style rules apply to those components that should have a similar style. Each style rule is designed using a special XML attribute style. Any changes made in the style rule will be automatically applied to all GUI components using this style. The following is a listing of an example styles.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <style name=" WelcomeTextView ">
      <item name="android:layout_width">wrap_content</item>
      <item name="android:layout_height">wrap_content</item>
 ...
  </style>
  <style name="ContactTextView">
...
  </style>
...
</resources>
```

An example of using a style rule in a control element located in the activity_main.xml activity file:

```
<TextView
style="@style/WelcomeTextView"
 ...
```

2.8 Comments and lack of preprocessors

Java supports three kinds of comments:

1. In the style of the C language (from "/ *" to "* /").

2. In the style of the C ++ language (from "//" to "\ n").

3. Special comments "for documentation" (from "/ **" to "* /").

Java does not have a preprocessor similar to the C++ preprocessor. And that means there are no directives such as #define, #include, #ifdef. In fact Java doesn't need them.

The equivalent of a constant designated in C / C ++ via #define is a variable designated in the Java class as static final, for example: java.lang.Math.PI. The advantages of this approach are: strict typing of constants and uniqueness of hierarchical names, which eliminates conflicts (the same PI cannot be redefined).

Macros (another use of #define) are replaced by the inline substitution implemented by the Java compiler for short methods automatically.

The unnecessity of # include is due to two factors. Firstly, a class file (with the extension .class) is at the same time both a class declaration and its definition (implementation), which means that there is no need of header h-files. Secondly, the standardization in the placement of class files in folders enables the Java interpreter to uniquely determine the location of the loaded class, which means that there is no need to include source files directly.

NOTE! Unfortunately, the lack of inclusion of fragments of the source text can make the source class files excessively large. This is especially true due to the fact that Java allows the definition of classes in a class, as discussed below.

Conditional compilation (# ifdef / # endif in C ++) in Java is done implicitly. The fact is that a normal Java compiler (for example, javac) determines sections of the source text that will never be executed, and ignores them without generating byte code for them. This means that the C / C ++ construct is in the following form:

```
#ifdef DEBUG
... debugging code ...
#endif
```

can be simulated in Java by construction:

```
private static final boolean DEBUG = true / false;
if ( DEBUG) {
... debugging code ...
};
```

Since DEBUG is a constant, the compiler knows before debugging whether debugging code will ever be executed or not.

Encoding characters, strings and identifiers (i.e. class names, variable methods) in Java are formed by 16-bit Unicode characters. This provides, for example, the ability to give classes and their members Russian names.

NOTE. Unicode representation of Latin characters is the same as their representation in ASCII and ISO8859-1 (Latin-1).

But, unfortunately, Unicode includes a Cyrillic encoding that matches the ISO8859-5 standard, which is not popular among us (although all UNIX systems support it).

An esc sequence \ u xxxx , where x is a hexadecimal digit, is used to represent Java characters that do not have a graphic representation (for example, due to the lack of appropriate fonts) . For example, \ u044E represents the lowercase Russian letter "ю ".

2. 9 Definition of the application menu: menu.xml file

The following is an example listing of an XML file to define one of the application's activities (view_contact_menu.xml) menu:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
        <item android:id="@+id/editItem"
                android:title="@string/menuitem_edit_contact"
```

```
            android:orderInCategory="1" android:alphabeticShortcut="e"
            android:titleCondensed="@string/menuitem_edit_contact"
            android:icon="@android:drawable/ic_menu_edit"></item>
        <item android:id="@+id/deleteItem"
            android:title="@string/menuitem_delete_contact"
            android:orderInCategory="2" android:alphabeticShortcut="d"
            android:titleCondensed="@string/menuitem_delete_contact"
            android:icon="@android:drawable/ic_delete"></item>
    </menu>
```

Menu XML files are located in the res / menu folder of the application, like other resource files. Each XML menu file contains a root <menu> tag, where the quantity of <item> tags are equal to menu items for implementation of this activity.

For each menu item, the value of the android : id attribute is determined, which enables programmatic interaction with the corresponding menu item. Other menu item attributes are listed in the following list:

1. Attributes android : title and android: titleCondensed. Using these attributes, the text displayed by the menu item is determined. The second attribute is applied if the text is too long to be displayed in the corresponding menu item.

2. Android : icon attribute . This attribute defines the image (drawable) that is displayed in the menu item above its text. In the examples of menu items three standard icons are used that are provided by the Android SDK. These icons are located in the data / res / drawable-hdpi folder of each version of the SDK. To refer to these icons in the XML markup, @android values are used: drawable / icon_name.

3. Android attribute : alphabeticShortcut. By means of this attribute, the letter is defined which is to be pressed by the user on the physical keyboard to select a menu item.

4. Android attribute : orderInCategory . This attribute defines the order in which menu items are displayed.

2.10 Conclusions to the second section

This section provides brief information about application development tools.

An analysis of the necessary software was carried out, and methods for creating the application were also determined.

The information considered will be taken into account when developing a ready-made mobile application.

3 IMPLEMENTATION OF THE "HANDBOOK FOR CELEBRATION" APPLICATION

3.1 Application Activity Implementation

When creating GUI screens, the Activity class is belonged and the View is used to interact with the user.

Each activity is a screen (similar to the form) that the application can display to users. The more complex the created application, the more screens (activities) will be required. When creating an application, you will need at least the initial (main) screen, which provides the basis for the user interface of the application. If necessary, this interface is supplemented by secondary activities designed to enter information, display it and provide additional capabilities. Starting (or returning from) a new activity leads to a "movement" between UI screens.

Most activities are designed to take advantage of the entire screen space, but you can also create translucent or floating dialog boxes.

If you want the theme to refer to the only seperate activity, but not to entire application, then the android: theme attribute must be added to the < activity > tag.

In most cases, there is no need to come up with your own styles and themes, since Android contains many of its own integrated themes. For example, you can use the Dialog theme to make the application screen look like a dialog box

<activity android:theme="@android:style/Theme.Dialog">

For a transparent screen, you can use the Translucent theme:

<activity android:theme="@android:style/Theme.Translucent">

If you like the topic, but still want to change some of your own themes , you just need to add the topic as a parent theme to your own topic. For example, I want to modify the standard Theme_Light theme to use my colors:

```
<color name="custom_theme_color">#b0b0ff</color>
<style name="CustomTheme" parent="android:Theme.Light">
<item
name="android:windowBackground">@color/custom_theme_color
</item>
<item
name="android:colorBackground">@color/custom_theme_color
</item>
</style>
```

Now in the manifest you can use your own style instead of Theme.Light :

```
<activity android:theme="@style/CustomTheme">
```

The following is a short list of properties that can be used to customize your own themes:

android: windowNoTitle - use true to hide the title.

android: windowFullscreen - use true to hide the status bar and free up space for the application.

android: windowBackground - a color resource or drawable for the background.

android: windowContentOverlay - drawable , which is drawn over the window contents. By default, this is the shadow of the status bar. You can use null (@ null in the XML file) to delete a resource

.

New styling features for applications for Android 5.0 (API level 21) and higher can be found at [12].

3.2. Markup code development strings.xml

The strings.xml file does not have a reference to the XML naming scheme in the Android repository, since it contains only resource definitions. It contains the parent <resources> tag and child <string> tags, each of which consists of a variable name in the name attribute and its value, such as: "Flower is", "Jewelry", "Toys", "For Parents".

```
<resources>
<string name="txtCredits">Support: <a href="http://www.stackoverflow.com">click here</a></string>
<string name="app_name">Handbook for Celebration</string>
<string name="navigation_drawer_open">Open navigation drawer</string>
<string name="navigation_drawer_close">Close navigation drawer</string>
<string name="nav_header_title">Handbook for Celebration </string>
<string name="nav_header_subtitle">android.studio@android.com</string>
<string name="nav_header_desc">Navigation header</string>
<string name="action_settings">Settings</string>

<string name="Flower">Flower</string>
<string name="Jewelry">Jewelry</string>
<string name="Toys">Toys</string>
<string name="ForParents"> For Parents</string>
<string name="Information">Information</string>
<string name="Advice">Advice</string>
<string name="Flower_1"> ***</string>
<string name="Flower_2">***</string>
```

```
<string name="Flower_3">***</string>
<string name="Flower_4">***</string>
<string name="Flower_5">***</string>
<string name="Flower_6">***</string>
</resources>
```

By clicking on each category a page opens. The user will be shown a list of topics where you can see information on clicking on each topic.

3.3 MainActivity.java code development

To create a new activity, the Activity class is inherited. Inside the class implementation, you must define the user interface and implement the required methods. The basic framework for the new activity is shown in the program code below:

```
package com.example.mynewapp;
import android.content.Intent;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import com.google.android.material.navigation.NavigationView;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import android.view.Menu;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
```

```
import android.widget.ListView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.Arrays;
```

Public class MainActivity extends AppCompatActivity and implements NavigationView.OnNavigationItemSelectedListener, DrawerLayout.DrawerListener methods:

```
{
  private DrawerLayout drawer;
  private ListView List;
  private String[] array;
  private ArrayAdapter<String> Adapter;
  private Toolbar toolbar;
  private int category_index;
}
```

Method onCreate() is called to create activities and uses a class object Bundle of packet android.os , which contains status and setting items UI activity as a binder (bundle) values and settings. An application can save its state if the operating system switches to the background, for example, if a user launches another application or receives a phone call.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
List = findViewById(R.id.ListView);
array = getResources().getStringArray(R.array.Flower);
```

```
Adapter = new ArrayAdapter<>(this,android.R.layout.simple_list_item_1,new ArrayList<String>(Arrays.asList(array)));
List.setAdapter(Adapter);
toolbar = findViewById(R.id.toolbar);
setSupportActionBar(toolbar);
drawer = findViewById(R.id.drawer_layout);
NavigationView navigationView = findViewById(R.id.nav_view);
navigationView.setNavigationItemSelectedListener(this);
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar, R.string.navigation_drawer_open, R.string.navigation_drawer_close);
drawer.addDrawerListener(toggle);
toggle.syncState();
navigationView.setNavigationItemSelectedListener(this);
List.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
Intent intent = new Intent(MainActivity.this, Text_Content_Activity_2.class);
intent.putExtra(   "category", category_index);
intent.putExtra(   "position", position );

startActivity(intent);
    }
    });

// ***
// ***
     }
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// ***
getMenuInflater().inflate(R.menu.main, menu);
toolbar.setTitle(R.string.Flower);
return true;
}

@Override
public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
int id = menuItem.getItemId();
if (id == R.id.id_Flower) {
toolbar.setTitle(R.string.Flower);
array = getResources().getStringArray(R.array.Flower); //***
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
category_index = 0;
}
if (id == R.id.id_Jewelry)
{
toolbar.setTitle(R.string.Jewelry);
array = getResources().getStringArray(R.array.Jewelry);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Jewelry_present ", Toast.LENGTH_SHORT).show();
category_index = 1;
}
if (id == R.id.id_Toys)
```

```
{
toolbar.setTitle(R.string.Toys);
array = getResources().getStringArray(R.array.Toys);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Toys ", Toast.LENGTH_SHORT).show();
category_index = 2;
}
if (id == R.id.id_ForParents)
{
toolbar.setTitle(R.string.ForParents);
array = getResources().getStringArray(R.array.ForParents);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
category_index = 3;

Toast.makeText(this, " For Parents ", Toast.LENGTH_SHORT).show();
}
if (id == R.id.id_Information)
{
toolbar.setTitle(R.string.Information);
array = getResources().getStringArray(R.array.Information);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Information ", Toast.LENGTH_SHORT).show();
category_index = 3;
   }
```

```
if (id == R.id.id_Advice)
    {
toolbar.setTitle(R.string.Advice);
array = getResources().getStringArray(R.array.Advice);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Advice ", Toast.LENGTH_SHORT).show();
category_index = 4;
}
//  ***
return true;
}

@Override
public void onDrawerSlide(@NonNull View drawerView, float slideOffset) {
}
@Override
public void onDrawerOpened(@NonNull View drawerView) {
}
@Override
public void onDrawerClosed(@NonNull View drawerView) {
}
@Override
public void onDrawerStateChanged(int newState) {
}
}
```

The body of the onCreate() method is composed of instructions for calling two methods. The first method call uses the keyword to pass the saveInstanceState object of the Bundle class to the onCreate() method of the base class (superclass)

android.app.Activity . A call to the second setContentView() method with a link to the XML file activity_main.xml, located in the layout folder, displays the content (Content View) of the main application activity on the screen . This instruction is as follows:

setContentView(R.layout.activity_main);

where R is the path to the folder that contains the layout folder of the project, for example:

C:\Users\Username\AndroidStudioProjects(workspace)\MyNewApp\res\
R.layout.activity_main

is converted to a file:

C:\Users\Username\AndroidStudioProjects\MyNewApp\res\layout\activity_main.xml

Thus, the setContentView() method loads the XML markup of the activity into a class derived from the base class Activity.

The second method of the onCreateOptionsMenu() class is used to create a menu for selecting activity options that opens when the <MENU> button is pressed on a mobile device. The activity menu is created from the main.xml XML file (located in the res/menu folder of the project) using the onCreateOptionsMenu() method with the menu argument of the android.view.Menu class .

The body of the onCreateOptionsMenu() method consists of one main statement and a return statement. The basic instructions for the object returned by calling method getRescelebources(), called activity, which takes two arguments:
menu , which was referred to the method onCreateOptionsMenu (Menu menu),

and

XML-file folder main.xml menu,

transmitted by the familiar structure R .menu.main denoting the file:

C:\Users\Username\AndroidStudioProjects\MyNewApp\res\menu\main.xml

Thus, method .inflate() takes an object of the Menu class and fills it, guided by the XML definition of the menu contained in the resource file. As soon as the activity menu is formed and ready to use, the method returns Android OS to true.

3.4 AndroidManifest.xml markup code development

Inheriting from the Activity class does not make the new activity automatically available. Since an activity can play various roles in an application, the Android platform requires some meta-information about the new activity. This meta-information must be provided by the XML < activity > tag in the AndroidManifest.xml application manifest file as below:

```
</manifest><?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.apress.helloworld" >
<application
android:allowBackup="true"
android:icon="@drawable/ic_launcher"
android:label="@string/app_name"
android:theme="@style/AppTheme" >
<activity
android:name=".MyActivity"
```

```
android:label="@string/app_name" >
<intent-filter>
<action android:name=
"android.intent.action.MAIN" />
<category android:name=
"android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
```

As you can see from the listing, XML tag <activity> designated a new activity providing class name, its title and how it should be presented to the user.

After the name of the project folder in Android studio is registered. After that there is a list of folders in the application that executes the icon, label, application name, content and theme .

Next is "Text_Content_Activity", allows the application to contain information by category.

Next comes the activity ‘android:name = ".MainActivity"’. Next comes ‘android:label = "@string/app_name"’ and ‘android:theme = "@style/ AppTheme.NoActionBar "’>. The "intent-filter" filter, with which the system will pass the implicit Intent object to the application only if it can go through one of your Intent filters.

3.5 Functionality description

The Mobile App “Handbook for the Holidays” represents select of best gift for any type of events and holidays to the user. The application is focused on holidays and debris that will provide useful information for them, which will help when choosing the best gift (fig. 3.1).

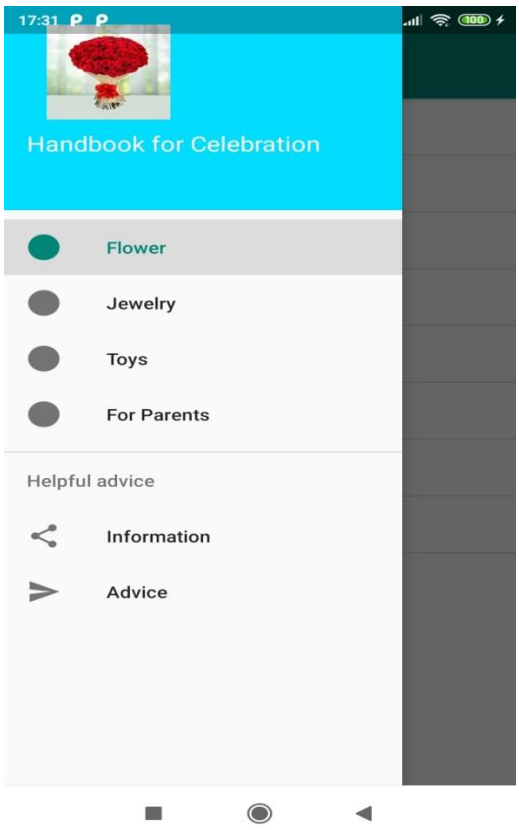

Figure 3.1 - The main screen menu

The application offers a choice of six categories: "Flower", "Jewelry", "Toys", "For Parents", "Information", "Advice".

After selecting any, the category by pressing you can see below the following screenshots (fig. 3.2, 3.3, 3.4).

On these pages, there are available different windows with lists of topics where you can see appropriate information on categories. Procedure is possible by clicking on each topic.

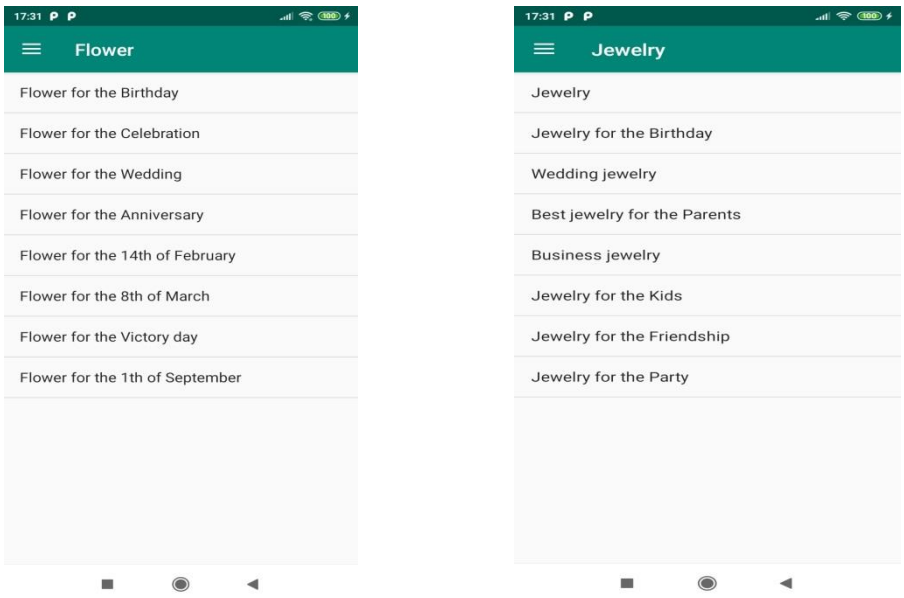

Figure 3.2 - “Flower” and “Jewelry” categories

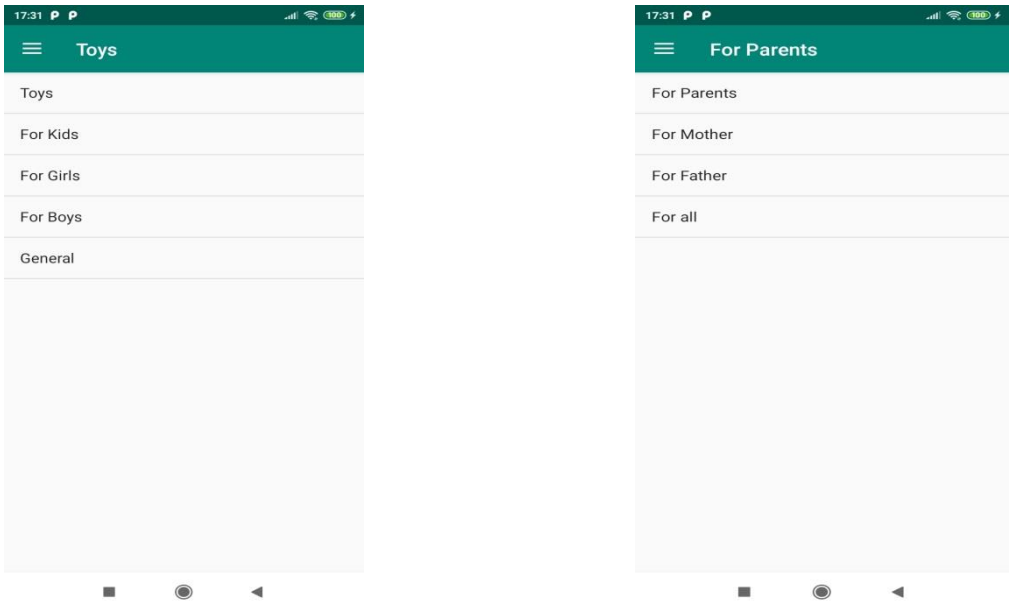

Figure 3.3 - "Toys" and "For Parents" categories

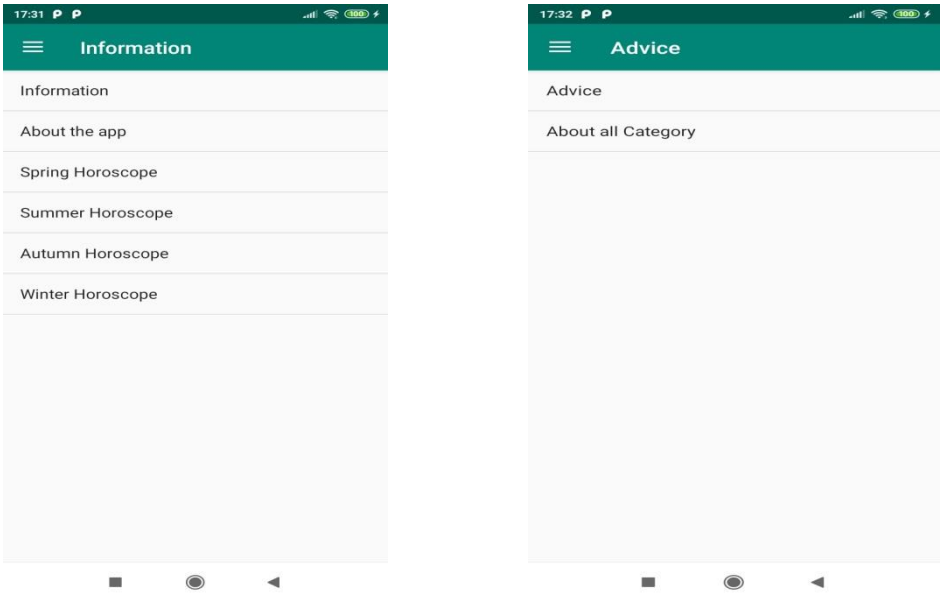

Figure 3.4 - "Information" and "Advice" categories

Below is an example of a screenshot of an application after selecting one of the events in the “Flower” category (fig. 3.5).

Figure 3.5 - Screenshot after selecting of the events in the “Flower” category

3.6 Conclusions to the third section

The third section describes the steps for creating an application, describes the interfaces and functions of the menu.

The section includes:

1. Description of application development.
2. Description of the interface with all categories.

The section also contains code fragments of the executed application pages and screenshots for each category.

4 OCCUPATIONAL HEALTH AND SAFETY

In this section the analysis of potential dangerous and harmful production factors, the reasons of fires are carried out. Measures to ensure occupational health and industrial sanitation are examined. Based on the analysis, safety measures and recommendations for fire prevention have been developed.

The aim of this bachelor work was to develop software and computer management system of online store.

Since design process was carried out by the computer, the analysis of potentially dangerous and harmful production factors are carried out for a personal computer on which the developed software and computer system will be developed and used.

4.1 General issues of labor protection

Working conditions at the workplace, safety of technological processes, machines, mechanisms, equipment and other means of production, the state of collective and individual protection used by the employee, as well as sanitary conditions must meet the requirements of labor protection regulations. In the " labor protection law" of Ukraine [13] stipulates that labor protection is a system of legal, socio-economic, organizational and technical, sanitary and hygienic and treatment-and-prophylactic measures and means aimed at preserving life, health and ability to work of human beings in the process of labor activity.

Working with computers change the physical and chemical factors of the environment: there is static electricity, electromagnetic radiation, changes in temperature and humidity, the level of oxygen and ozone in the air. Improper organization of the workplace leads to the general and local tension of the muscles of the neck, corpora, upper extremities, spinal curvature and the development of osteochondrosis.

4.2 Analysis of working conditions

Work on creating a website will take place in the apartment building where the student lives. One person is enough for this job who needs a workplace with a desktop computer.

4.2.1 Premises requirements

The geometric dimensions of the room are listed in table. 4 .1.

Table 4.1 - Dimensions of the room

Name	Value
Length, m	5
Width, m	3
Height, m	2
Area, m^2	15
Volume, m^3	30

According to GSN 3.3.6.042 -99 [18] the size of the area for one workplace of a personal computer operator must be at least 6 square meters. m, and the volume - not less than 20 cubic meters. m. Therefore, this room fully complies with these standards.

To provide the required lighting, the room must have a window and a system of steady light installed on the ceiling. According to the fire safety requirements there should be the quickest way out from the room through a window with ventilation. In the room there is always a blanket as well that can be used in case of fire.

4.2.2 Requirements for the organization of the workplace

To compare the characteristics of the workplace accordance with regulations and main requirements for organization of workplace according to STATE STANDARDS 3.3.2.007 -98 [16] (Table. 4.2) and the appropriate values of workplace the following full compliance is given hereinafter.

Table 4.2 - Characteristics of the workplace

The name of the parameter	In fact Value	Normative value
Height of a working surface, mm	700	680 - 800
Height of space for legs, mm	650	not less than 600
Width of space for legs, mm	630	not less than 500
Depth of space for legs, mm	670	not less than 650
Seat surface height, mm	420	400 - 500
Seat width, mm	400	not less than 400
Seat depth, mm	400	not less than 400
Height of a surface of a back, mm	500	not less than 300
Width of a basic surface of a back, mm	470	not less than 380
The radius of curvature of the back in the horizontal plane, mm	400	400
Distance from the eyes to the display screen, mm	720	700 - 800

The office is located in a single-floor building with a volume of 54 m3, area - 1 5 m2.

The temperature in the room during the year varies between 18-24 ° C, relative humidity - about 50%. The room ventilation system is unorganized, and the heating is provided individually.

The placement of windows provides natural light with a coefficient of natural light of at least 1.5%, and the total artificial lighting, which is carried out using a single fluorescent lamp, provides a level of 55 illumination of at least 200 lux. According to the degree of fire safety, the room belongs to category B.

4. 3 Industrial sanitation

Based on the analysis of hazardous and harmful factors during production (operation), to meet requirements of fire safety employees must be provided with sufficient lighting, air ventilation, grounding.

4.3.1 Analysis of hazardous and harmful factors in the production (operation) of the product

Analysis of hazardous and harmful production factors is performed in tabular form (Table 4.3). Work related to EOP with VDT, including equipped with computers workplaces with VDT and CHP, is performed with the implementation of NPAO 0.00- 7.15 -1 8 [17], which sets safety requirements for the equipment of workers places to work with the use of computers with VDT and CHP. Most projects are performed in offices or in other premises where a variety of electrical equipment is used, including personal computers (PCs) and peripherals. The main performance characteristics of a personal computer are:

operating voltage U=+220V +-5%;

operating current I=2A;

power consumption P=350 W.

Workplaces must meet the requirements for sanitary rules and regulations for working with visual display terminals of electronic computers, approved by the resolution of the Chief State Sanitary Doctor of Ukraine from 10.12.98 № 7 [16].

Table 4.3 - Analysis of hazardous and harmful production factors

Dangerous harmful production factors	Sources of factors	Quantitative assessment	Regulations
1	2	3	4
Physical			
With high or low humidity	Operation of computers, printers, scanners or server equipment for work	3	[18]
Increased voltage of the electrical network, the short circuit of which can occur through the human body	Operation of computers, printers, scanners or server equipment for work	2	[21]
Insufficient lighting of the work area	Violation of hygienic parameters of the production environment	1	[19]
Physiological:			
Neuropsychological overloads	formulation of the topic; information research of subject area; work performance documantation	4	[16]
- Physical (static-sitting);	Violation of organization of labour conditions of the work place (continuous service)	2	[14] [17]

4. 3 .2 Fire safety

Fire safety when using a computer is provided by:

1) fire prevention system;

2) fire protection system;

3) organizational and technical measures.

Potential sources of ignition can be:

1) sparks and arcs of short circuit;

2) electric spark when closing and opening circuits;

3) overheating from prolonged overload;

4) open fire and combustion products;

5) the presence of substances heated above the autoignition temperature;

6) bit static electricity.

The causes of possible fires and fires can be:

1) malfunction of the electrical installation;

2) design defects of the equipment;

3) short circuit in electrical networks;

4) ignition of combustible materials in the immediate vicinity of the electrical installation.

4. 3 .3 Electrical safety

The following electrical safety requirements are met at the workplace: PCs, peripherals and maintenance equipment, electrical wires and cables in terms of performance and degree of protection correspond to the zone class according to PUE (rules of electrical installations), have short-circuit current protection equipment and other emergency modes. The power supply line for PC power supply, peripherals and maintenance equipment is made as a separate group three-wire network, by laying phase, neutral operating and neutral protective conductors. The neutral protective conductor is used for grounding (zeroing) of electric receivers. Plug connections and

electrical sockets in addition to the contacts of the phase and neutral conductors have special contacts for connecting the neutral protective conductor. The mains sockets for powering personal PCs are laid on the floor next to the walls in accordance with the approved equipment placement plan and technical characteristics of the equipment. Metal pipes and flexible metal sleeves are grounded. Protective grounding includes grounding devices and a conductor that connects the grounding device to the equipment to be grounded - the grounding conductor.

4.4 Hygienic requirements for the parameters of the production environment

4. 4 .1 Microclimate

The microclimate of working premises is the climate of the internal environment of these premises, which is determined by the combination of temperature, humidity, speed of air movement acting on the human body. In this room, work is performed while sitting and does not require dynamic physical exertion, it corresponds to the category of work 1a. Therefore, the optimal values for temperature, relative humidity and air mobility for the specified workplace correspond to DCN 3.3.6.042-99 [18] and are given in table. 4.4:

Table 4.4 - Norms of the microclimate of the working area of the object

Period of the year	Category of works	Temperature C 0	Relative humidity, %	Air velocity, m / s
Cold	Easy-1a	22-24	40-60	0,1
Warm	Easy-1a	23-25	40-60	0,1

4. 4 .2 Lighting

Lighting calculation .

For industrial and administrative premises the light factor is accepted not less than -1/8, in household - 1/10:

$$\sqrt{a^2 + b^2} \cdot Sb = \left(\frac{1}{8} \div \frac{1}{10}\right) \times Sn, \quad (4.1)$$

where S_b – area of window opening, m^2;

S_n – floor area, m^2.

$S_n = a \cdot b = 5 \cdot 3 = 15\ m^2$,

$S_{age} = 1/8 \cdot 15 = 1{,}875\ m^2$.

We accept a window with the area $S = 1{,}875\ m^2$.

The calculation of artificial lighting is made according to the coefficients of light flux utilization, which determine the flux required to create a given illuminance in general uniform illumination. The calculation of the number of lamps n is made by the formula (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M} \quad (4.2)$$

Where n - the number of lamps;

E - normalized illumination of the work surface, determined by the norms - 300 lux ;

S - illuminated area, m^2 ($S = 15\ m^2$);

Z - correction factor of the lamp (1.15 for incandescent lamps and DRL; 1.1 for fluorescent lamps) , take equal to 1.1 ;

K - stock factor, taking into account the reduction of illumination during operation - 1.5 ;

U - utilization factor, depending on the type of lamp, the index of the room, etc. , 0.575M ;

M is the number of fluorescent lamps in the lamp , 1 unit ;

F - luminous flux - 54 00lm (for LB-80) .

According to the requirements of DBN B.2.5-28: 201 8 [19] visual work in the operation of a personal computer refers to the work of medium visual accuracy (category IV, subsection "a"), the illumination of the workplace of the computer operator must be 300 lux. As a result of substitution of numerical values in the formula (4.2) number of fixtures:

$$n = \frac{300 \cdot 15 \cdot 1.15 \cdot 1.5}{5400 \cdot 0.575 \cdot 1} = 2.5 \approx 3$$

We accept a lighting installation consisting of three luminaires equipped with LB type lamps (one - 80 W) with a luminous flux of 5400 lm.

4. 5 Ventilation

In the room where there are computers, air exchange is realized by means of the natural organized ventilation (ventilation shafts), ie at V of the room> 40 m3 on one worker natural ventilation is allowed. This method provides Prov and for the right amount of fresh air that is defined in the building regulations. Ventilation of the room should also be carried out, depending on weather conditions, the duration should be at least 10 minutes. The best air exchange is carried out by through ventilation.

To maintain the optimal temperature in the room in accordance with the requirements of DBN V 2.5-67: 2013 [20] there is central heating and ventilation. In the warm season, air conditioning is used.

4. 6 Measures to organize the production environment and prevent emergencies

1 . Safety measures during the operation of personal computers and peripherals include:

- proper organization of the workplace and compliance with optimal modes of work and rest when working with a PC;

- operation of certified equipment;

- observance of electrical safety measures;

- ensuring optimal parameters of the microclimate;

- ensuring rational lighting of the workplace (illumination of the workplace did not exceed 2/3 of the normal illumination of the room);

- when arranging a room for working with a PC, it is necessary to provide supply and exhaust ventilation or air conditioning:

2. Safety measures during the operation of other electrical appliances include compliance with the following rules:

- constantly monitor the condition of the mains, switchboards, switches, sockets, lamp sockets, as well as mains power cables with which electrical appliances are connected to the mains;

- constantly monitor the serviceability of the insulation of the power grid and network cables, preventing their operation with damaged insulation;

- do not pull on the mains cable to pull the plug out of the socket;

- do not cover with furniture, various inventory switches, plug sockets;

- do not connect several powerful electrical devices to one socket at the same time, which can cause excessive heating of conductors, destruction of their insulation, melting and ignition of polymeric materials;

- do not leave switched on electrical appliances unattended;

4. 6 .1 Calculation of protective grounding (ensuring the electrical safety of the building)

According to the classification of premises according to the degree of danger of electric shock NPAOP 40.1-1.01-97 [19], the room in which all work is carried out belongs to the first class (without increased danger). When the use of electrical supply voltage 36 V, 220 V and 360 V. The resistance circuit in the ground should be no more than 4 ohms.

The sequence of calculation.

1.The required resistance of artificial grounding conductors $R_{dm.з.:}$

$$R_{шт.з.} = \frac{R_{\delta.} \cdot R_{пр.з.}}{R_{пр.з.} - R_{\delta.}}, \quad (4.3)$$

where $R_{рг.3.}$ – resistance of natural grounding conductors;

R is the allowable ground resistance.

If natural earthing conductors are absent, then $R_{шт.з.}=R_{д.}$

Substituting the numerical values in formula (4.3), we obtain:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2. Grounding resistance largely depends on the resistivity of the soil ρ, Ohm·m. The approximate value of the resistivity of the clay is taken as ρ=40 Ohm•m (tabular value).

3. The calculated specific resistance of the soil, $P_{розр}$, Ohm·m, is determined respectively for vertical grounding conductors $P_{розр.в}$, and horizontal $P_{розр.г.}$, Ohm·m by the formula:

$$P_{розр.} = \psi \cdot \rho \quad (4.4)$$

where ψ – the seasonality factor for vertical grounding and climatic zones with normal soil moisture, taken for vertical grounding $\rho_{розр.в} = 1.7$ and horizontal $\rho_{розр.г} = 5.5$ ohm·m.

$P_{розр.в.} = 1.7 \cdot 40 = 68$ Ом/м

$P_{розр.г.} = 5.5 \cdot 40 = 220$ Ом/м

4. The current resistance of the vertical grounding conductor $R_в$, Ом, by (4.5) is calculated..

$$R_в = \left(\ln\frac{2 \cdot 1в}{dcT} + \frac{1}{2} \cdot \ln\frac{4 \cdot t + 1в}{4 \cdot t - 1в}\right)) \quad (4.5)$$

Where $l_в$ – the length of the vertical grounding (for pipes - 2−3 m; $l_в$=3 m);

$d_{ст}$ – diameter of the rod (for pipes - 0,03−0,05 m; $d_{ст}$=0,05 m);

t – is the distance from the ground to the middle of the ground, which is determined by f. (4.6);

$$t = h_в + \frac{1_в}{2} \quad (4.6)$$

where $h_в$ – the depth of laying vertical grounding (0.8 m); then

$$t = 0.8 + \frac{3}{2} = 2{,}3 \text{ m}$$

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln\frac{2 \cdot 3}{0.05} + \frac{1}{2} \cdot \ln\frac{4 \cdot 2.3 + 3}{4 \cdot 2.3 - 3}\right) = 18.5 \text{ Om}$$

5. The theoretical number of vertical grounding n pieces is determined, without taking into account the utilization factor ηв pcs:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18.5}{4} = 9.25 \quad (4.7)$$

6. The required number of vertical grounding conductors is determined taking into account the utilization factor $n_в$, pcs:

$$n_в = \frac{2 \cdot R}{R_д \cdot n_в} = \frac{2 \cdot 18.5}{4 \cdot 0.57} = 16.2 \approx 16 \quad (4.8)$$

7. The length of the connecting tape of the horizontal grounding l_c,

$$lc = 1.05 \cdot Lв \cdot (nв - 1) \quad (4.9)$$

where $l_в$ – the distance between the vertical grounding, (take l in= 3m);

nb – the required number of vertical grounding.

$l_c = 1.05 \cdot 3 \cdot (16 - 1) \approx 48m$

8. The resistance to current flow of the horizontal grounding Rg, Ohm is determined:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot 1c} \cdot \ln \frac{2 \cdot 1^2}{d_{см} \cdot h} \quad (4.10)$$

where $d_{см}$ – is the equivalent diameter of the strip with a width b, $d_{см}$ =0,95b, b = 0,15 m;

h_r – depth of laying of horizontal grounding conductors (0.5 m);

1_c – length of the connecting tape of the horizontal grounding conductor 1_c, m

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0.95 \cdot 0.15 \cdot 0.5} = 8.1 \text{ Om}$$

9. The coefficient of use of the horizontal grounding ηc. is determined. according to the required number of vertical grounding nv.

The coefficient of use of the connecting strip η_c=0,3 (tabular value).

10. The resulting resistance of the ground electrode is calculated taking into account the connecting strip:

$$R_{заг} = \frac{R_в . R_г}{R_в \cdot n_c \cdot R_г \cdot n_в \cdot n_в} \leq R_д, \quad (4.11)$$

Conclusion: this protective grounding will ensure the electrical safety of the building, as the condition is fulfilled: R total <4 Ohms, namely:

$$R_{заг} = \frac{18.5 \cdot 8.1}{18.5 \cdot 0.3 \cdot +8.1 \cdot 16 \cdot 0.57} = 1.9 \leq R_д$$

4. 7 Conclusions to the fourth section

As a result of the work, an analysis of working conditions, harmful and dangerous factors faced by the workers was made. The parameters and certain characteristics of the work room in the proposed project written in the qualification work were defined, it is described what measures need to be taken to ensure that the room meets the necessary standards and is comfortable and safe for the worker.

Recommendations on the organization of the workplace, as well as important information on fire and electrical safety. The dimensions of the room and the values of temperature, humidity and air mobility, the required number of lamps and other parameters, the value of which affects the working conditions of the worker, as well as instructions on labor protection, safety when working on a computer .

CONCLUSIONS

The purpose of the graduation project is to develop a Android application "Handbook for Celebration" using the Android Studio software. The application implements the choice of the best gift for any type of holiday by category, horoscope, advice, information about the application and a convenient interface.

An analysis of the necessary software was carried out, and methods for creating the application were also determined. The information considered will be taken into account when developing a ready-made mobile application.

Developed Android Application “Handbook for Celebration” represents select of best gift for any type of events and holidays to the user. The application is focused on holidays and debris that will provide useful information for them, which will help when choosing the best gift.

In this work it was analyzed working conditions, harmful and dangerous factors faced by the workers was made. The parameters and certain characteristics of the work room in the proposed project written in the qualification work were defined, it is described what measures need to be taken to ensure that the room meets the necessary standards and is comfortable and safe for the worker.

THE LIST OF REFERENCES

1. Paul Deitel, Harvey Deitel, Alexander Wald. Android 6 for Programmers: An App-Driven Approach, 3/E. - Deitel & Associates, Inc., New York, 2016. – 460 pp.

2. Dave MacLean, Satya Komatineni, Grant Allen. Pro Android 5. - Springer Science+Business Media, New York, 2015. – 813 pp.

3. Robert Love. Linux System Programming, Second Edition. - O'Reilly Media, Sebastopol, 2013. – 456 pp.

4. The Way of Android OS - https://www.npd.com/wps/portal/npd/us/about-npd/search/?query=Android.

5. Brett Spell. Pro Java 8 Programming. - Springer Science+Business Media, New York, 2015. – 695 pp.

6. Neil Smyth. Android Studio 3.6 Development Essentials – Java Edition. - Payload Media, Inc.

7. Одиночкіна С. В. Основи технологій XML (Навчальний посібник) / Санкт-Петербург: Національний дослідницький університет інформаційних технологій, механіки та оптики, 2013. – 57 с.

8. Anders Göransson. Efficient Android Threading. - O'Reilly Media, Inc., Gravenstein Highway North, Sebastopol, 2014. – 279 pp.

9. Holiday Calendar 2020 - https://play.google.com/store/apps/details?id=com .apps.calendar&hl=ru.

10, Wallance Jackson. Android Apps for Absolute Beginners; Covering Android 7. – Lompoc, California, USA, 2017. - 499 pp.

11. Robert Sedgewick, Kevin Wayne. Introduction to Programming in Java. Second Edition. - Pearson Education, Inc., New York, 2017. – 780 pp.

12. Material Design for Android - https://developer.android.com/guide/topics /ui/look-and-feel.

13. Law of Ukraine "On labor protection". Come into force by the Resolution of VP № 2695-XII of 14.10.92, VVP, 1992, ? 49, Article 669. - Access mode: https://zakon.rada.gov.ua/laws/show/2694-12

14. Law of Ukraine "On Compulsory State Social Insurance against Accidents at Work and Occupational Diseases That Caused Disability". Order of December 21, 2000 N 2180-III. Access mode: https://zakon.rada.gov.ua/laws/show/1105-14

15. Code of Labor Laws of Ukraine. Approved by Law № 322-VIII of 10.12.71 VVP, 1971. Access mode: https://zakon.rada.gov.ua/laws/show/322-08

16. State sanitary rules and norms of work with visual display terminals of electronic computers GSanPIN3.3.2.007-98. Approved by the Resolution of the Chief State Sanitary Doctor of Ukraine of December 10, 1998 N 7. Access mode: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

17. NPAOP 0.00-7.15-18 "Requirements for safety and health of workers when working with screen devices". Registered at the Ministry of Justice of Ukraine on April 25, 2018 at № 508/31960. Access mode: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

18. Sanitary norms of microclimate of production premises of DCN 3.3.6.042-99. Resolution No. 42 of December 1, 1999. Access mode: https://zakon.rada.gov.ua/rada/show/va042282-99

19. DBN V.2.5-28: 2018 "Natural and artificial lighting". Valid from 02/28/2019. Access mode: www. URL: https://dbn.co.ua/load/normativy/dbn/dbn_v_2_5_28/1-1-0-1188

20. DBN.2.5 -67.2013 "Heating, ventilation and air conditioning". Valid from 01.01.2014. Access mode: https://drive.google.com/file/d/1yoIHk5OZJ7vPvPbzmhjFAX7DTrH2H3Bo/view

21. NPAOP 40.1-1.01-97 "On approval of the Rules of safe operation of electrical installations". Registered in the Ministry of Justice of Ukraine on January 13, 1998 for the number 11/2451 Access: https://zakon.rada.gov.ua/laws/show/z0011-98.

Appendix A

Program listings

File listing MainActivity.java

```
package com.example.mynewapp;
import android.content.Intent;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import com.google.android.material.navigation.NavigationView;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import android.view.Menu;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.Arrays;
```

```
public class MainActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener, DrawerLayout.DrawerListener
{

private DrawerLayout drawer;

private ListView List;

private String[] array;

private ArrayAdapter<String> Adapter;

private Toolbar toolbar;

private int category_index;


@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

List = findViewById(R.id.ListView);

array = getResources().getStringArray(R.array.Flower);

Adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1,new
ArrayList<String>(Arrays.asList(array)));

List.setAdapter(Adapter);

toolbar = findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

drawer = findViewById(R.id.drawer_layout);

NavigationView navigationView = findViewById(R.id.nav_view);

navigationView.setNavigationItemSelectedListener(this);

ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar,
R.string.navigation_drawer_open, R.string.navigation_drawer_close);

drawer.addDrawerListener(toggle);
```

```
toggle.syncState();
navigationView.setNavigationItemSelectedListener(this);
List.setOnItemClickListener(new AdapterView.OnItemClickListener()

{

@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
Intent intent = new Intent(MainActivity.this, Text_Content_Activity_2.class);
intent.putExtra(   "category", category_index);
intent.putExtra(   "position", position );

startActivity(intent);

}
});

// ***
// ***

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// ***
```

```
getMenuInflater().inflate(R.menu.main, menu);
toolbar.setTitle(R.string.Flower);
return true;

}

@Override
public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
int id = menuItem.getItemId();

if (id == R.id.id_Flower) {
toolbar.setTitle(R.string.Flower);
array = getResources().getStringArray(R.array.Flower); //***
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
category_index = 0;

}
if (id == R.id.id_Jewelry)
{
toolbar.setTitle(R.string.Jewelry);
array = getResources().getStringArray(R.array.Jewelry);
Adapter.clear();
Adapter.addAll(array);
```

```
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Jewelry_present ", Toast.LENGTH_SHORT).show();
category_index = 1;

}

if (id == R.id.id_Toys)
{
toolbar.setTitle(R.string.Toys);
array = getResources().getStringArray(R.array.Toys);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Toys ", Toast.LENGTH_SHORT).show();
category_index = 2;

}
if (id == R.id.id_ForParents)
{
toolbar.setTitle(R.string.ForParents);
array = getResources().getStringArray(R.array.ForParents);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
category_index = 3;
```

```
Toast.makeText(this, " For Parents ", Toast.LENGTH_SHORT).show();
}
if (id == R.id.id_Information)
{
toolbar.setTitle(R.string.Information);
array = getResources().getStringArray(R.array.Information);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Information ", Toast.LENGTH_SHORT).show();
category_index = 3;

}
if (id == R.id.id_Advice)
{
toolbar.setTitle(R.string.Advice);
array = getResources().getStringArray(R.array.Advice);
Adapter.clear();
Adapter.addAll(array);
Adapter.notifyDataSetChanged();
Toast.makeText(this, " Advice ", Toast.LENGTH_SHORT).show();
category_index = 4;
```

```
}

//  ***
return true;

}
```

File listing Text_Content_Activity.java

```
package com.example.mynewapp;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
public class Text_Content_Activity extends AppCompatActivity {
private int category = 0;
private int position = 0;
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.text_content);
reciveIntent();
}
```

```
private void reciveIntent()
{
Intent i = getIntent();
if(i != null)
category = i.getIntExtra("category",0);
position= i.getIntExtra("position" , 0);
}
}
```

File listing Text_Content_Activity_2.java

```
package com.example.mynewapp;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.widget.TextView;
public class Text_Content_Activity_2 extends AppCompatActivity {
TextView textView;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.text_content);
textView = (TextView) findViewById(R.id.text_main_content);
textView.setMovementMethod(new ScrollingMovementMethod());
```

}

}

File listing strings.xml

<resources>

<string name="txtCredits">Support: click here</string>

<string name="app_name">Handbook for Celebration </string>

<string name="navigation_drawer_open">Open navigation drawer</string>

<string name="navigation_drawer_close">Close navigation drawer</string>

<string name="nav_header_title">Handbook for Celebration </string>

<string name="nav_header_subtitle">android.studio@android.com</string>

<string name="nav_header_desc">Navigation header</string>

<string name="action_settings">Settings</string>

<string name="Flower">Flower</string>

<string name="Jewelry">Jewelry</string>

<string name="Toys">Toys</string>

<string name="ForParents"> For Parents</string>

<string name="Information">Information</string>

<string name="Advice">Advice</string>

<string name="Flower_1">***</string>

<string name="Flower_2">***</string>

<string name="Flower_3">***</string>

<string name="Flower_4">***</string>

<string name="Flower_5">***</string>

<string name="Flower_6">***</string>

</resources>

File listing arrays.xml

<?xml version="1.0" encoding="utf-8"?>

<resources>

<string-array name="Flower">

<item> Flower for the Birthday </item>

<item> Flower for the Celebration </item>

<item> Flower for the Wedding </item>

<item> Flower for the Anniversary</item>

<item> Flower for the 14th of February</item>

<item> Flower for the 8th of March </item>

<item> Flower for the Victory day</item>

<item> Flower for the 1th of September </item>

</string-array>

<string-array name="Jewelry">

<item> Jewelry </item>

<item> Jewelry for the Birthday </item>

<item> Wedding jewelry </item>

<item> Best jewelry for the Parents </item>

<item> Business jewelry </item>

<item> Jewelry for the Kids </item>

<item> Jewelry for the Friendship </item>

<item> Jewelry for the Party </item>

</string-array>

<string-array name="Toys">

<item> Toys </item>

<item> For Kids </item>

<item> For Girls </item>

<item> For Boys </item>

<item> General </item>

</string-array>

<string-array name="ForParents">

<item> For Parents </item>

<item> For Mother </item>

<item> For Father </item>

<item> For all </item>

</string-array>

```
<string-array name="Information">
<item>    Information</item>
<item>    About the app </item>
<item>    Spring Horoscope</item>
<item>    Summer Horoscope </item>
<item>    Autumn Horoscope </item>
<item>    Winter Horoscope </item>
</string-array>

<string-array name="Advice">
<item>   Advice  </item>
<item>   About all Category  </item>

</string-array>

</resources>
```

File listing activity_main_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
tools:showIn="navigation_view">
```

```
<group android:checkableBehavior="single">
<item
android:id="@+id/id_Flower"
android:icon="@mipmap/ic_fl"
android:title="@string/Flower" />
<item
android:id="@+id/id_Jewelry"
android:icon="@mipmap/ic_jw"
android:title="@string/Jewelry" />
<item
android:id="@+id/id_Toys"
android:icon="@mipmap/ic_toys"
android:title="@string/Toys" />
<item
android:id="@+id/id_ForParents"
android:icon="@mipmap/ic_parents"
android:title="@string/ForParents" />
</group>

<item android:title="Helpful advice">
<menu>
<item
android:id="@+id/id_Information"
android:icon="@drawable/ic_menu_share"
```

```
android:title="@string/Information"  />
<item
android:id="@+id/id_Advice"
android:icon="@drawable/ic_menu_send"
android:title="@string/Advice" />
</menu>
</item>

</menu>
```

File listing text_content.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:orientation="vertical" android:layout_width="match_parent"
android:layout_height="match_parent">

<ScrollView
android:layout_width="match_parent"
android:layout_height="214dp"
android:layout_marginStart="8dp"
android:layout_marginLeft="8dp"
```

```
android:layout_marginTop="8dp"

android:layout_marginEnd="8dp"
android:background="@drawable/side_nav_bar">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="horizontal">

<ImageView
android:layout_width="3300dp"
android:layout_height="match_parent"
android:layout_weight="1"
android:src="@drawable/flower" />

</LinearLayout>
</LinearLayout>
</ScrollView>
```

```
<TextView
android:id="@+id/text_main_content"
android:layout_width="match_parent"
android:layout_height="434dp"
android:scrollbars = "vertical"
android:text="@string/Flower_1" />

<LinearLayout
android:layout_width="match_parent"
android:layout_height="100dp"
android:layout_marginTop="10dp"
android:background="@drawable/side_nav_bar"
android:orientation="horizontal"></LinearLayout>

</LinearLayout>
```

File listing AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.mynewapp">

<application
android:allowBackup="true"
```

```
android:icon="@mipmap/ic_launcher"
android:label="Handbook for Celebration"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".Text_Content_Activity_2"></activity>
<activity
android:name=".MainActivity"
android:label="Handbook for Celebration"
android:theme="@style/AppTheme.NoActionBar">
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>

</manifest>
```

Appendix B

Presentation slides

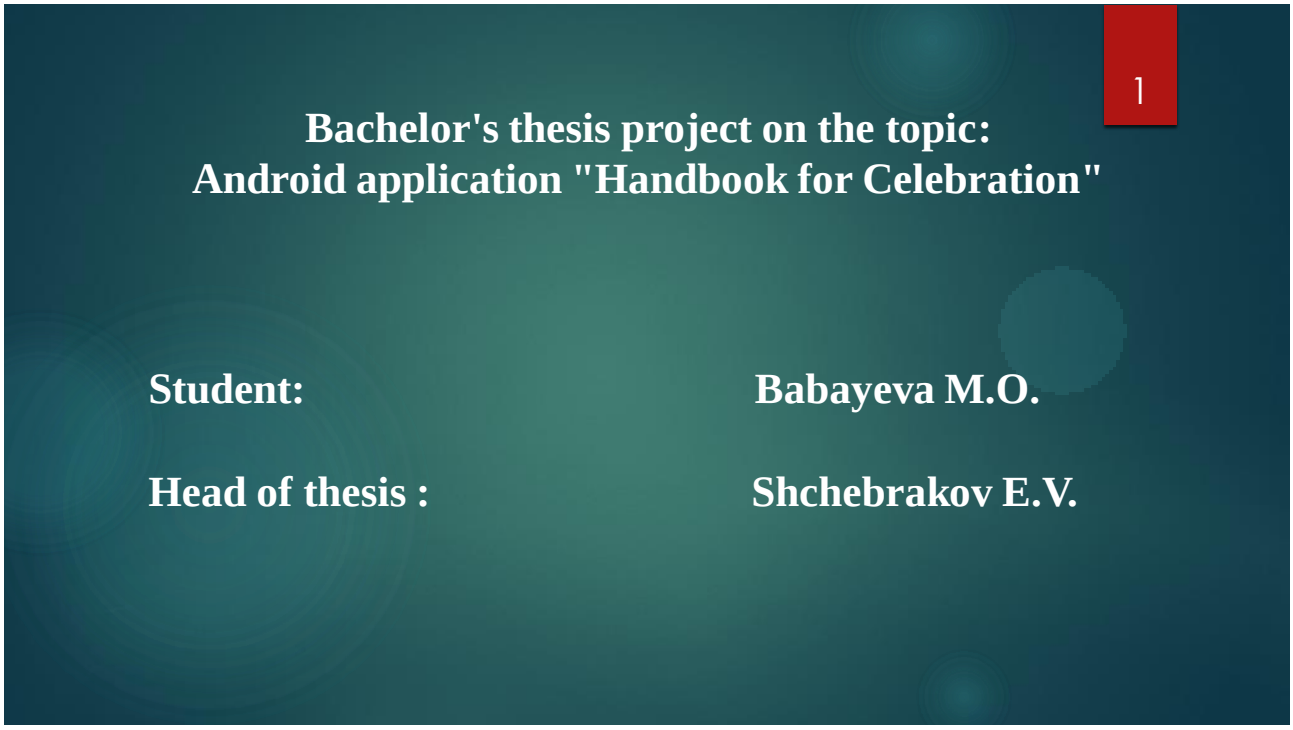

Figure B.1 - Main slide

Figure B.2 – Goal of the project

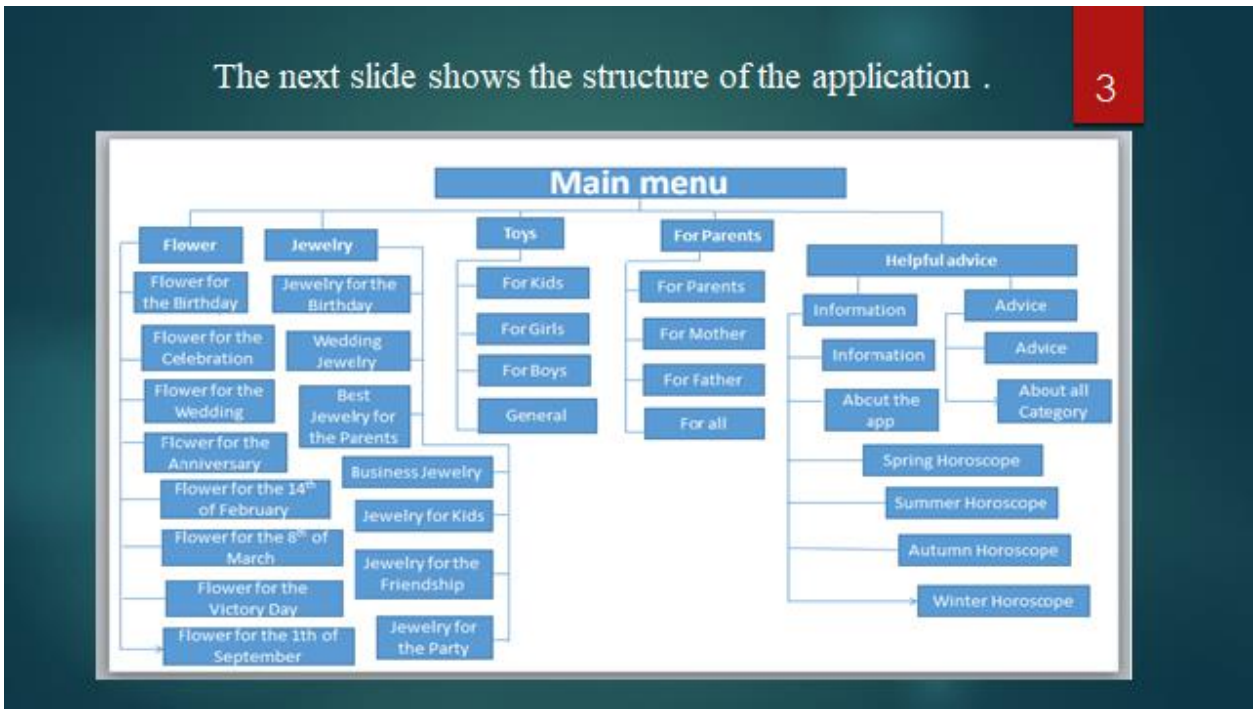

Figure B.3 - Structure

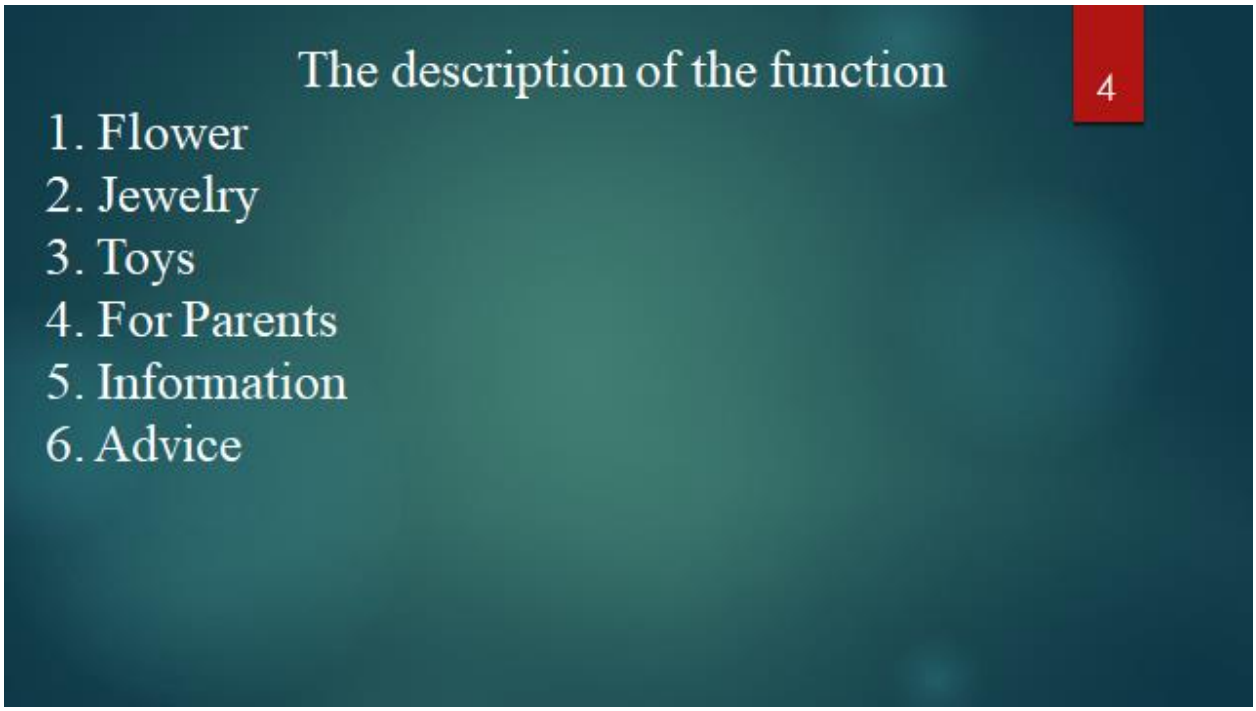

Figure B.4 - Function description

Figure B.5 - string.xml code fragment

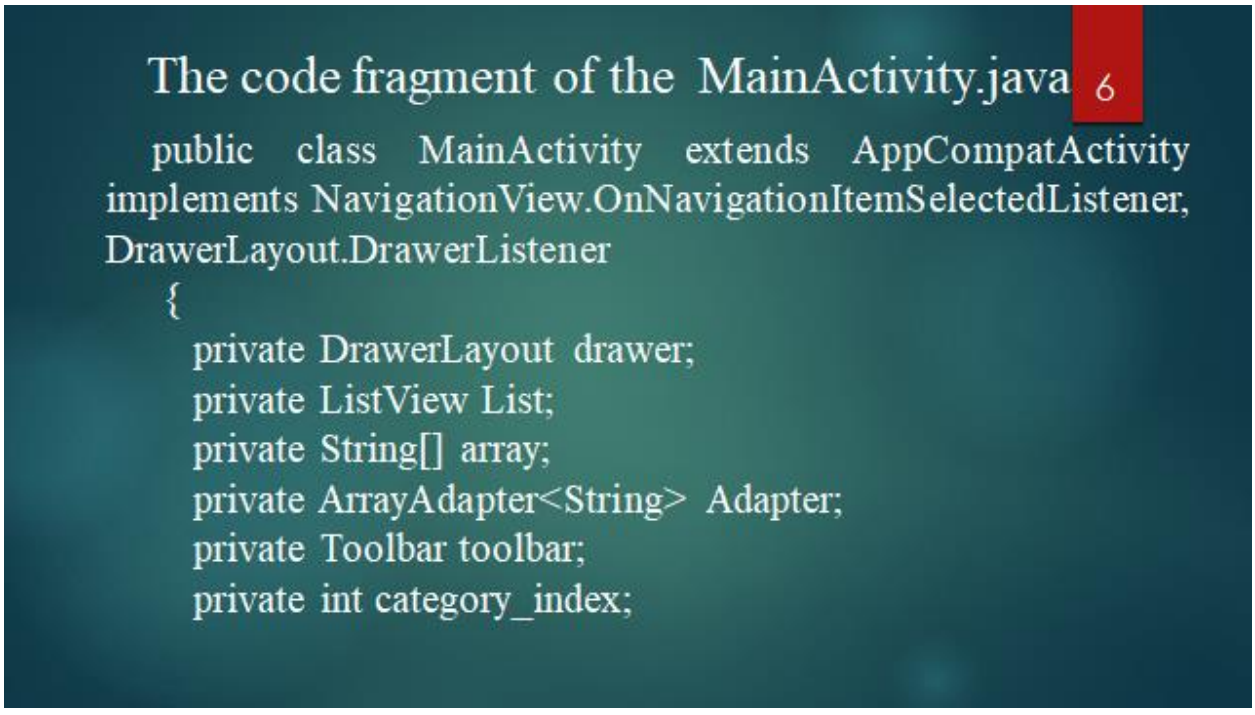

Figure B.6 – MainActivity.java code fragment

```
@Override
    public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
        int id = menuItem.getItemId();

        if (id == R.id.id_Flower) {
            toolbar.setTitle(R.string.Flower);
            array = getResources().getStringArray(R.array.Flower);
            Adapter.clear();
            Adapter.addAll(array);
            Adapter.notifyDataSetChanged();
category_index = 0;

        }
        if (id == R.id.id_Jewelry)
        {
            toolbar.setTitle(R.string.Jewelry);
            array = getResources().getStringArray(R.array.Jewelry);
            Adapter.clear();
            Adapter.addAll(array);
            Adapter.notifyDataSetChanged();
            Toast.makeText(this, " Jewelry_present ", Toast.LENGTH_SHORT).show();
            category_index = 1;
        }
```

7

Figure B.7 – Code continue fragment MainActivity.java

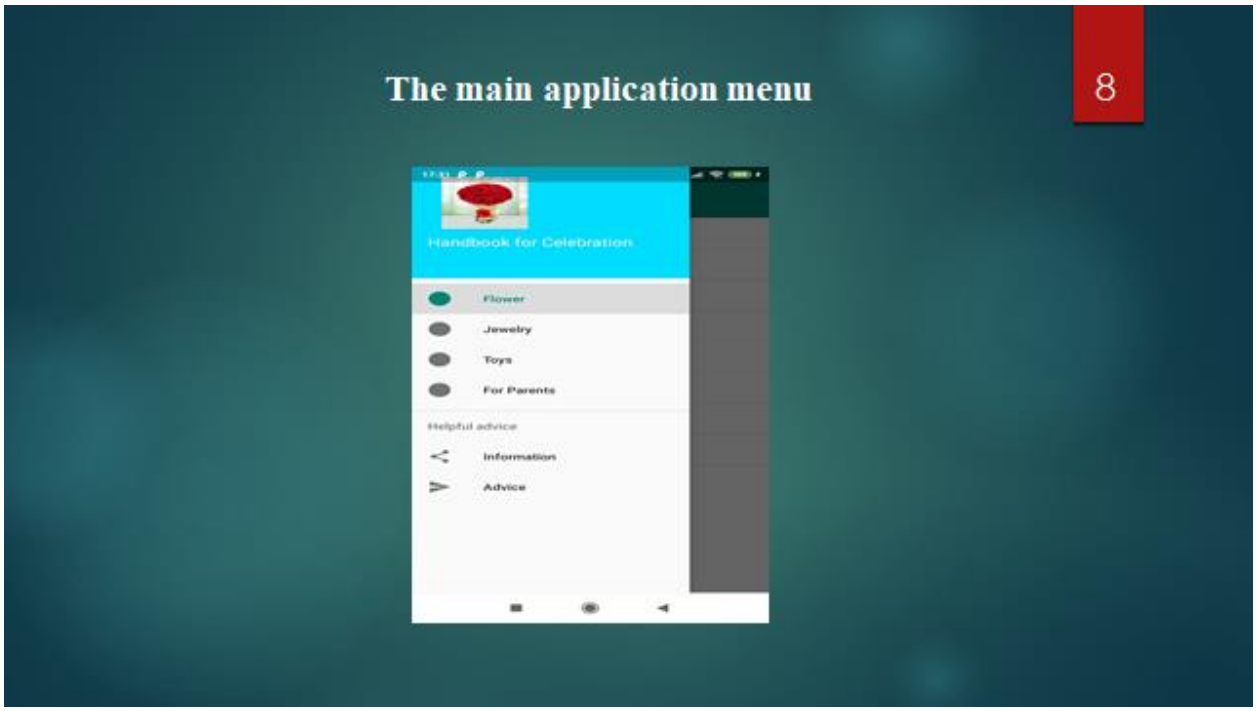

Figure B.8 – The main application menu

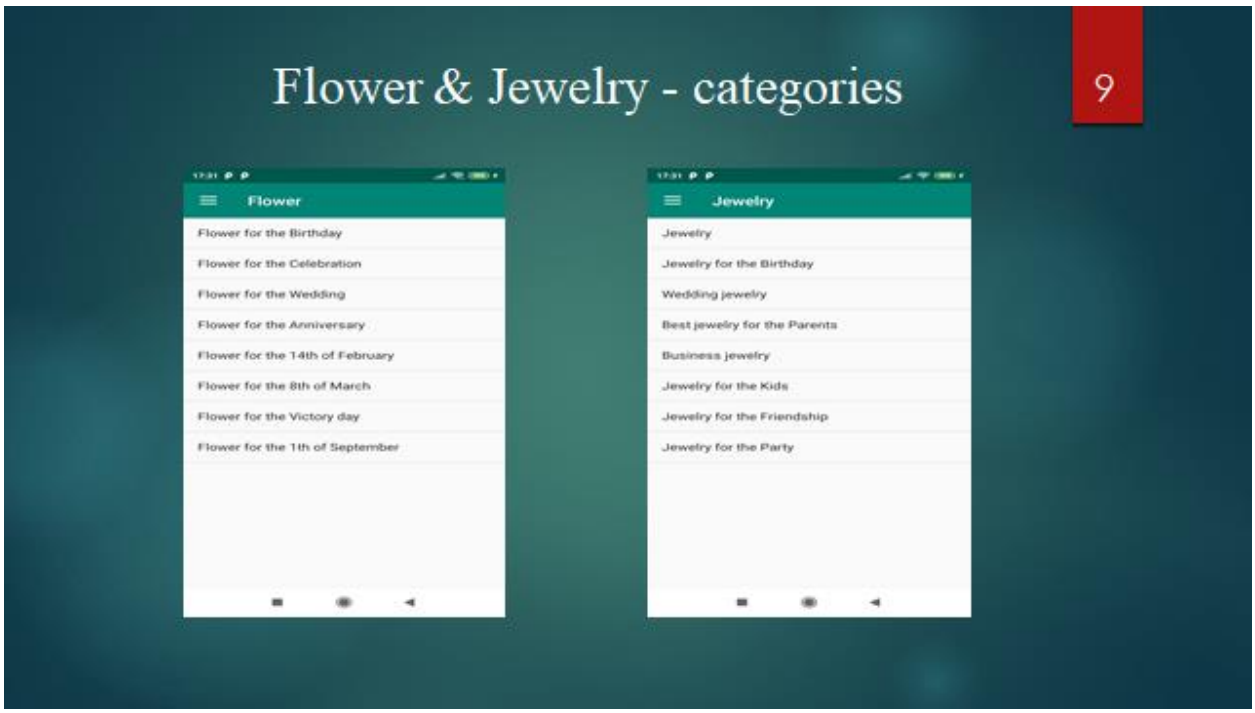

Figure B.9 - Flower & Jewelry – categories

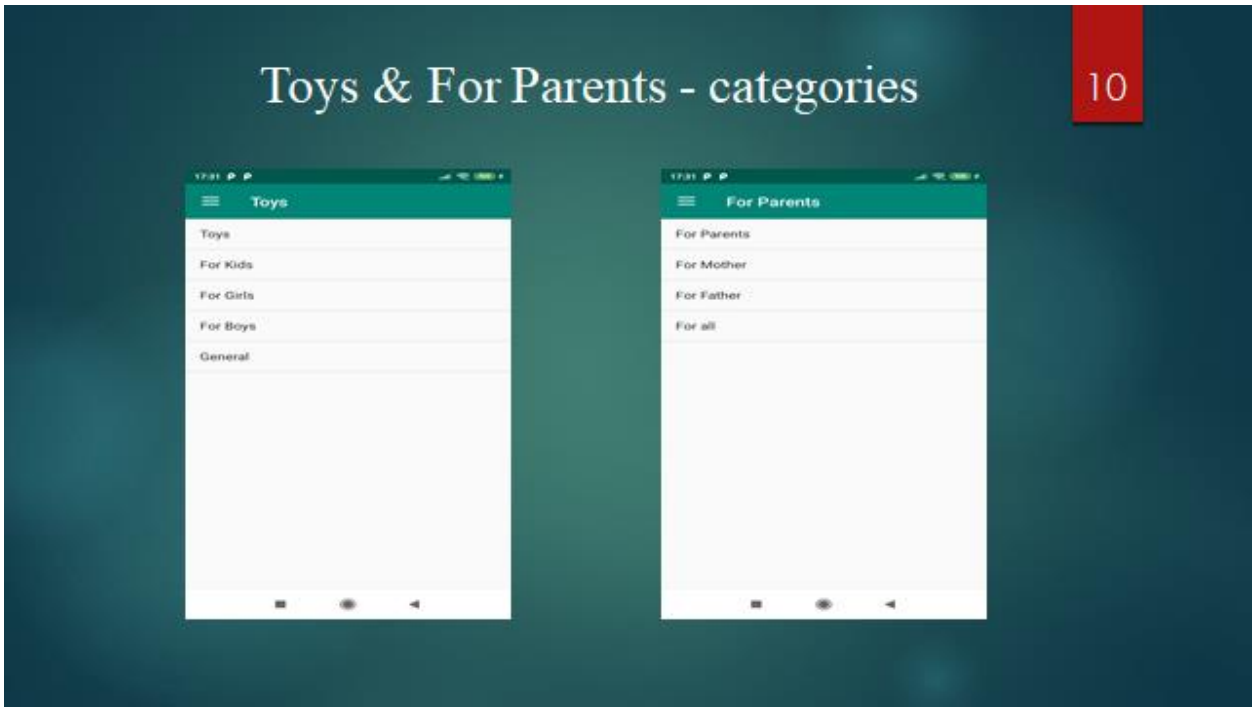

Figure B.10 – Toys & For Parents – categories

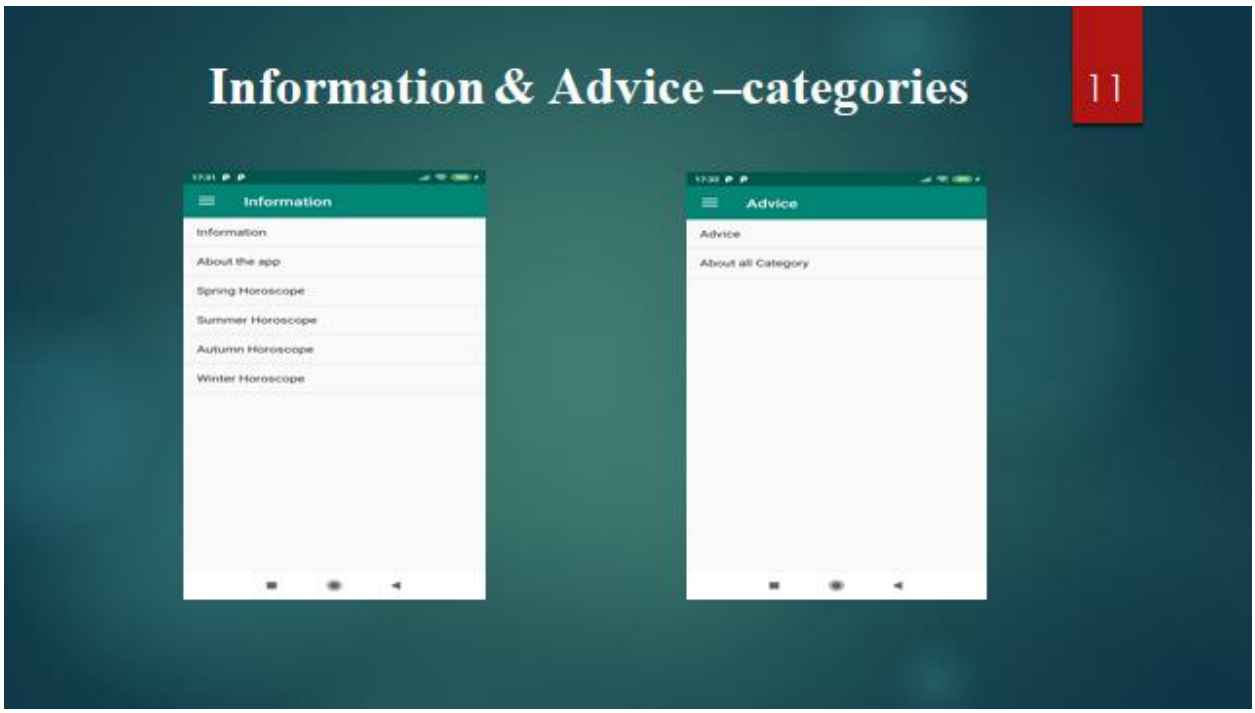

Figure B.11 – Information & Advice – categories

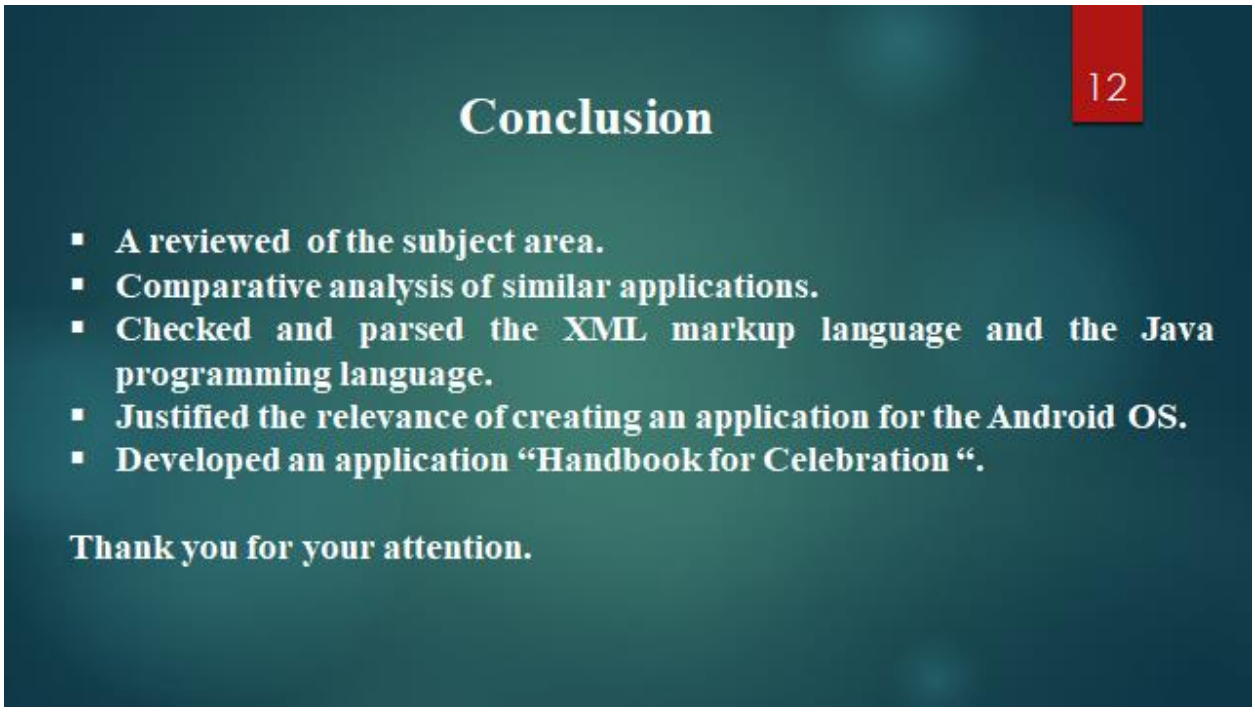

Figure B.12 – Conclusion