FRI-ONLINE-1-MIP-02

AN ANDROID APP "FOR TOURISTS BY TOURISTS" [2]

Prop. Assist. Valentin Velikov, PhD
Department of Informatics and Information Technologies,
University of Ruse
Tel.: +359 886 011 544
E-mail: vvelikov@ami.uni-ruse.bg

BSc. Petar Velikov
Department of Computer Science,
University of Roehampton, London
Phone: +447760191655
E-mail: trading@tradingforex.club

Abstract: *The paper reviews some Android technologies for working with the Internet, remote databases (big nonSQL database, as FireBase for example), threads, dynamic structures and more. The aim of the created practical application is to develop, acquire and test the corresponding technologies.*

It was necessary to explore and study these techniques and programming paradigms in connection with the creation and development of a lecture course.

„For tourists by tourists" – the idea of the developed system is that once in a given city, the users/tourists can browse and rate showplaces and landmarks, as well as offer new ones to the administrators.

Keywords: *Android, Software Engineering, Information systems, Tourism*

INTRODUCTION

Like the leading operating systems, Android is constantly updated and developed. Both new applications and new opportunities for their creation appear. Many of them are from external companies, and are accessible through libraries. Some of them Google acquires, and includes them in the development tools (API).

In connection with a lecture course, some of these possibilities had to be explored, used and demonstrated. For this purpose, the practical application "For tourists by tourists" was developed - a mobile application (client) for tourist attractions in a given locality.

EXPOSITION

Main goals set during the development of the application:

1. Building a mobile application for business purposes using the android platform.
2. Apply understanding of mobile user experience UX and implementation of accessibility features.
3. Identify properties and capabilities of modern mobile devices and the specific issues relating to software development for them.
4. Demonstrate cross platform choices and mobile application implementation techniques.
5. Explore new UI tools, such as Constraint Layout, RecyclerView, RatingBar, WebView and more.
6. Working with FireBase (a big, remote, non-SQL Database);
7. Working with threads when accessing the Database, making requests in the conditions of multithreading;

[2] Докладът е представен на конференция на Русенския университет на 13 ноември 2020 г. в секция Математика информатика и физика с оригинално заглавие на български език: ПРИЛИЖЕНИЕ ЗА ANDROID „ОТ ТУРИСТИ ЗА ТУРИСТИ".

8. Comparison and evaluation of the used dynamic structures for data storage in the RAM of the device.

To achieve these goals, has been developed the application "For tourists by tourists" (Fig.1). It has been decided that the database will be FireBase - Google's Cloud Platform. Its full administration should be carried out by an administrator in the respective municipality.

Idea for the application: Arriving in the city, the tourist can install the application by downloading it, for example. from the site of the municipality or from the Google store. The client (tourist) creates a new object (eg natural landmark), uploads a photo, makes a description (if desired), links to it (optional), uploads a location, gives a rating (rating) and the new object immediately becomes available to all who use the application.

Unregistered users can only view and rate the sights, and those wishing to register to the application can create / add new ones. The system administrator (for the application) can place or change a link or update the contents of the database.

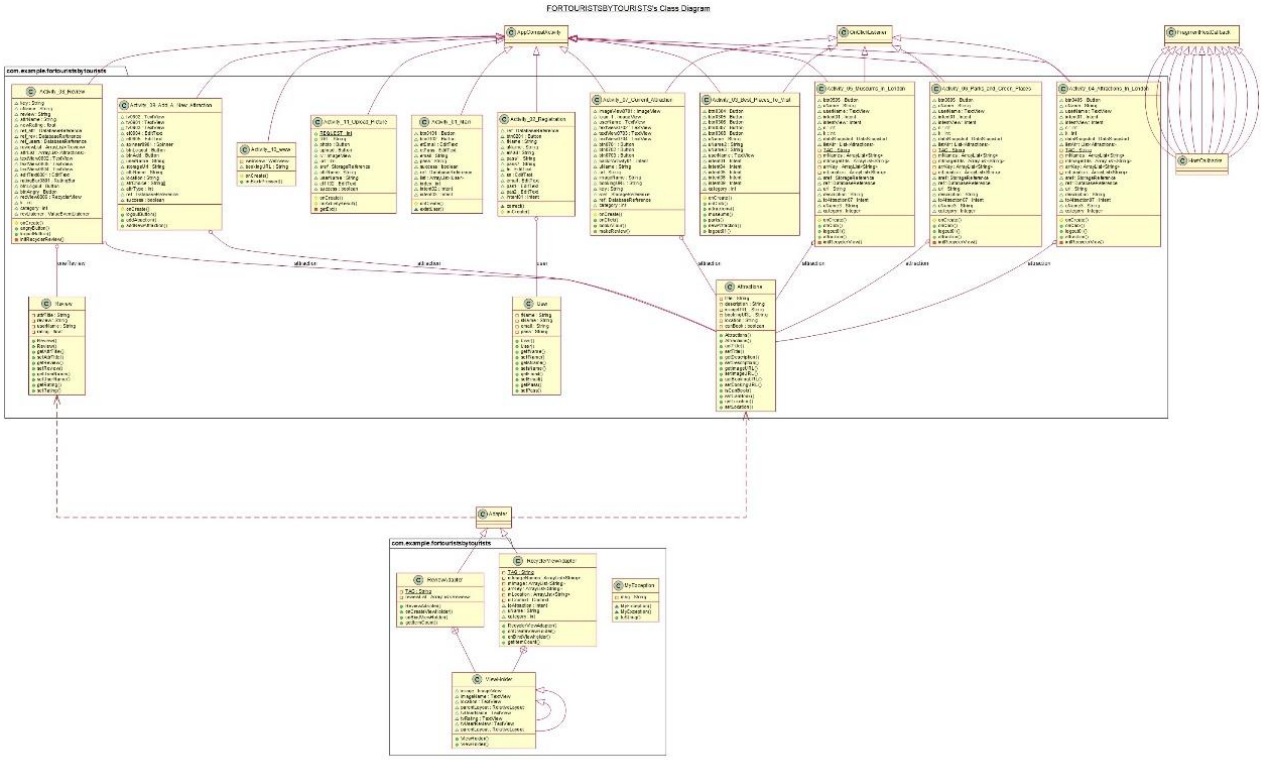


Fig. 1. Application class-diagram

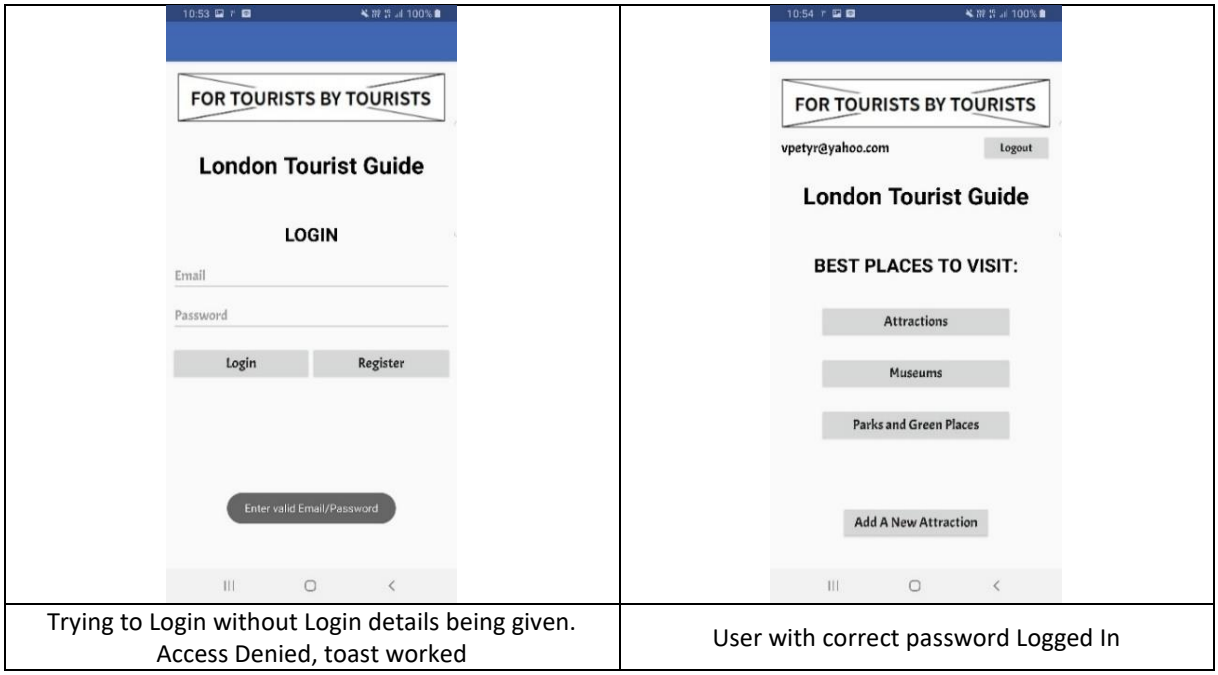
The solution is to link the information system with attractions in London - due to its popularity as a tourist destination, well-developed infrastructure, the availability of sites for most attractions and more.

Application Development

Perhaps the basic and most important rule for the designer. The product should be simple and clear, which doesn't mean that the design should be boring. The peak of mastery is to convey the message of a great idea in a simple and understandable language.

As a main rule for the design of all **activities** it was used the simplicity and similarity, helping the users actively move from slide to slide and changing pages without confusing themselves

An application starts with a registration page which allow tourists to register. First field - email address which must be unique and will be the login name, after that firstname, surname and password.



Trying to Login without Login details being given. Access Denied, toast worked	User with correct password Logged In

Fig. 2. LOGIN Activity

Tourists can be able to login to their accounts using their email address and password. It was created a function that checks the content of the corresponding fields fo correctness, and in case of correctness connects toFireBase. Users who are not registered denied access to posting reviews and booking features of the app. Once logged in they are taken to a welcome dashboard where options to view the categories of:

- **Attractions**
- **Museums**
- **Parks and green spaces**

To ensure that attractions are viewed, information about them is transferred from the database (*FireBase*) to several dynamic lists (*ArrayList*) - for images, for text description, for location, for the key value. There was a problem developing the application: sometimes it works, sometimes an error occurs. It turns out that internally the system makes the request to the database in a separate (implicit, internal) thread, and when the data is delivered before the user scrolls, everything works. However, if the data is not delivered - a problem arises. This necessitated the development of a logic to ensure the transfer of the database before the initialization of the *ReciclerView* component, ie. the initialization must take place after the implicit thread has finished its work.

A significant problem with the use of external (so far) dialog components was the lack of sufficient documentation and examples of their use on official Google sites. This necessitated a search for information and tutorials by developers on forums and YouTube [8] [9].

Using a standalone class (*ViewHolder*) to visualize an object, in combination with the accompanying xml (*layout_listitem.xml*), allows easy rearrangement of elements in an object (image, title, explanatory text), without affecting the functionality of the product.

A custom class (*MyException*) has been developed for handling and visualizing custom exceptions.

Sightseeing (via *RecyclerView*), a landmark can be selected. Described through the *Attractions* class, this ensures a uniform presentation of all landmarks. For each of them, *FireBase* provides a name, image, description, category, link to a web page.

A. Attractions

Here is a list of the attractions (Fig.3) listed in alphabetical order with optionally a thumbnail image with the attraction name. E.g:

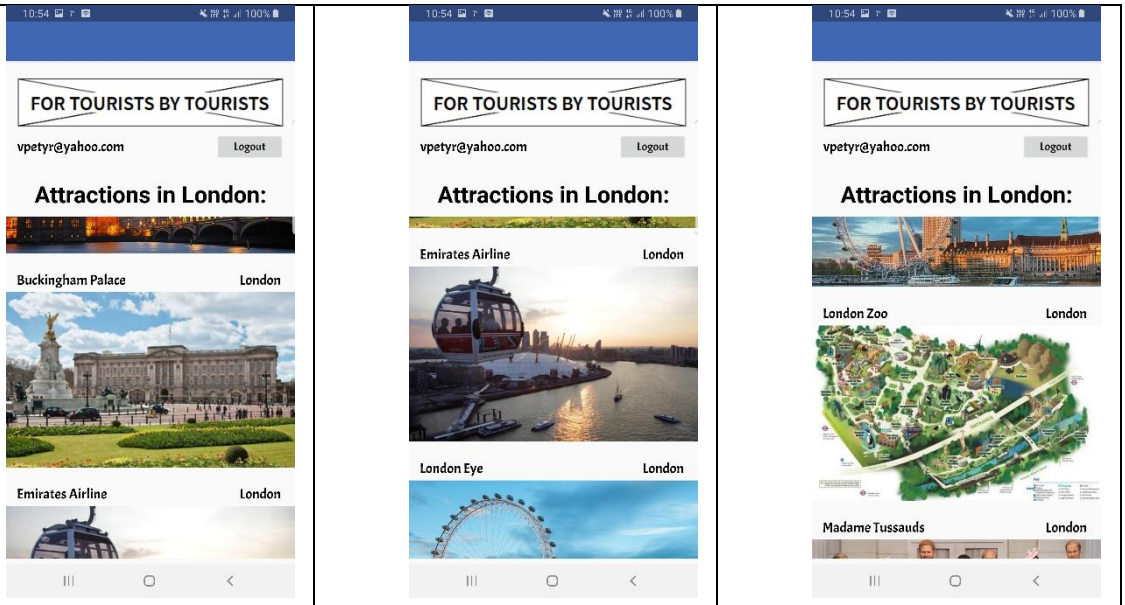


Fig. 3. Attractions in London – LIVE, scrolling up/down, using Recycler View
The database information is being transferred (attractions and picturesURL) in ArrayLists and is being given to the RecyclerView adapter for visualization

The *RecyclerView* element was used to view the existing attractions. At the time the app was created, it was an external component, and an additional library had to be included to use it. This functionality is currently acquired by Google, and is now available as a *widget* to the Android API.

RecyclerView is used to display the contents of dynamic lists (when the number of items displayed is not known in advance). This widget offers scrollable elements (objects) whose content is provided by a dynamic list.

Current Attraction Page	Current Attraction Page WIREFRAME

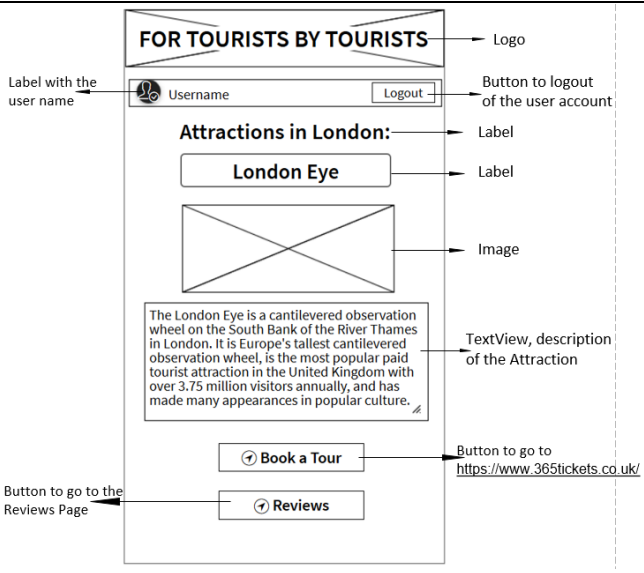


Fig. 4. One activity has been used for all of the three attraction categories. The number of the category is given from the previous activity. The category being chosen will select which table in the database the program will use. The picture will be loaded from the Firebase Storage.

An additional class (*ViewHolder*) describes the visualization of each displayed item in the list, and information about the object itself is provided by the *Attractions* class.

These attractions are not manually hard written to the activity or fragment. They are dynamically read from the database and then set on the activity (*activity* – an Android software

component with view/display). When the user clicks on one of the attractions, a new activity with details of the attraction then be displayed with options to write reviews, view existing reviews, book a tour. E.g. (Fig. 4):

Scrolling up/down using a ReciclerView to read the reviews left by users (Fig.5). We can see the username, rating being displayed and the reviews being left and written in the Firebase Database.

Tourists can read reviews from other tourists and can make (registrated only) reviews of the locations and rate it.

Fig. 5. Reviews left by users

If the user chooses to book a tour, a new activity will display taking the user to https://www.365tickets.co.uk/ where the user can book a ticket for the tour. E.g. if a user were to click book on Buckingham palace the new activity should link specifically to: https://www.365tickets.co.uk/buckingham-palace?q=buckingha from where they can book their ticket.

This is the *WebView*

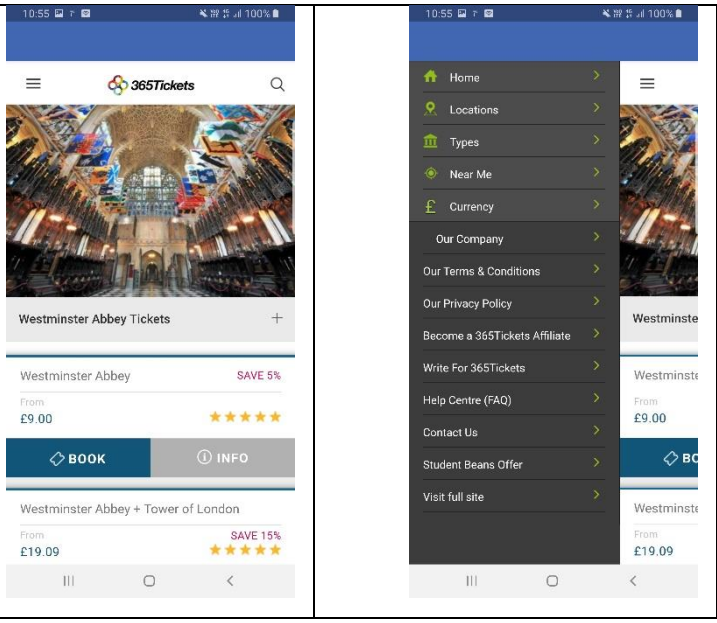


Fig. 6. The WebWiew widget is used to link to the booking site

B. Museums

Similar to the activity above, this activity shows list of museums in London. More of 10 museums are shown. Tourists can post reviews and view reviews of other tourists as well.

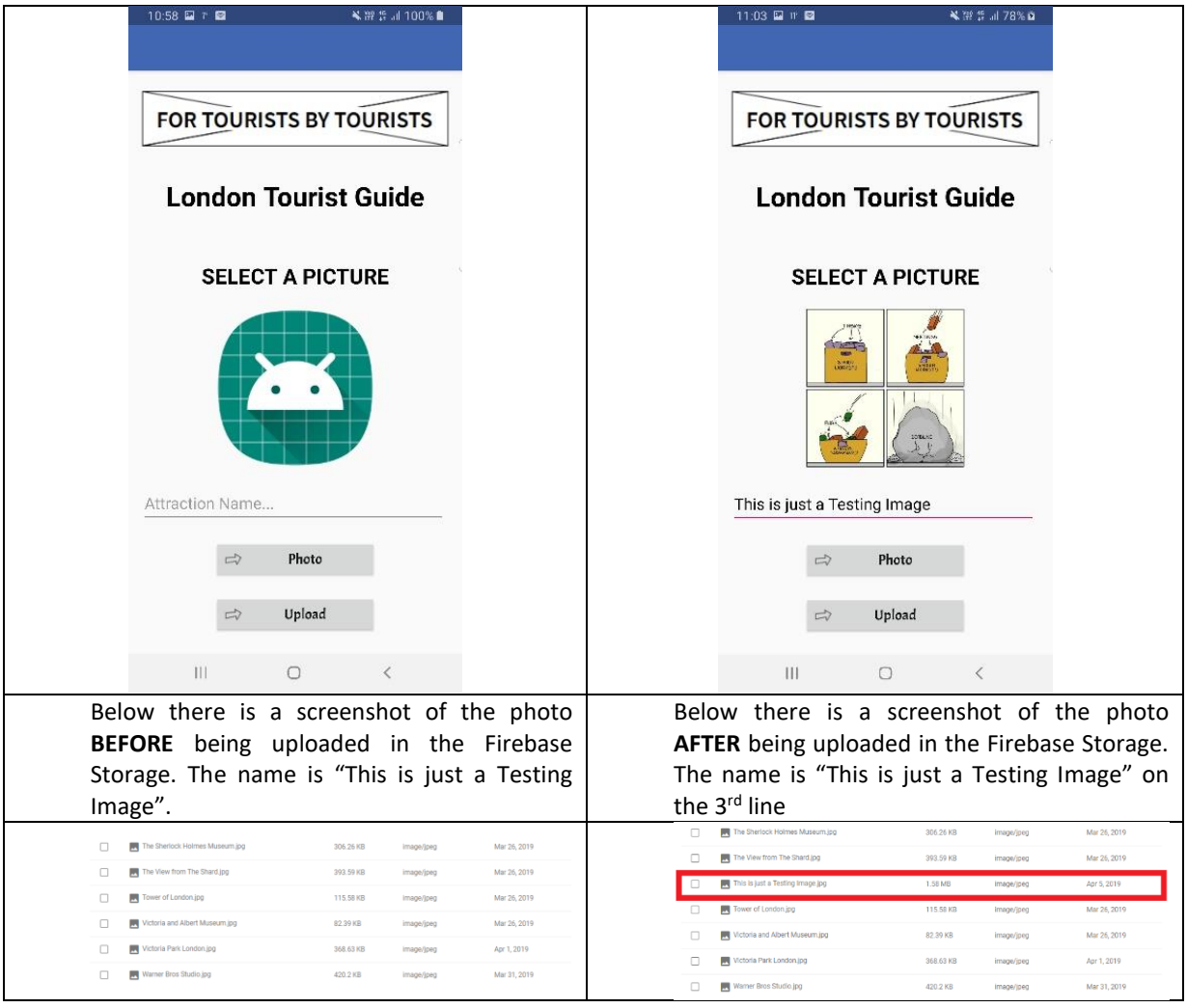
C. Parks and green spaces

Similar to the activity above this activity show list of parks. For green spaces thy are purely tourist led in terms of listed content. Tourists can add new attractions that they think are nice and that are not already in the database, with the name of the place, picture, location and description are being uploaded to the database. Little known green spaces such as for example:

- A butterfly house at Clissold Park.
- Camley Street natural park.
- Barnsbury wood nature reserve.
- Abney Park Cemetary.

Tourists can make reviews of the locations and read reviews from other tourists.

In each attraction category registered users can add new attractions (Fig. 7). First they need to upload a photo with a name in the database, and then they can add a short description. Attraction name should be typed first, which will be used for a key generation in the Firebase Database and as a name of the picture as well in the Firebase Storage.



Below there is a screenshot of the photo **BEFORE** being uploaded in the Firebase Storage. The name is "This is just a Testing Image".

Below there is a screenshot of the photo **AFTER** being uploaded in the Firebase Storage. The name is "This is just a Testing Image" on the 3rd line

Fig. 7. Add new attractions

Database connectivity

For the task delivery the team use Google's Cloud Platform – a Firebase (Fig. 9, Fig.10) based application allowing to developers to synchronize the data automatically between mobile and web versions. The Firebase Database is a non SQL Database. The application uses the JSON format for the data structure description (Fig. 9).

The data connectivity used is similar to SQLite (Fig. 8).

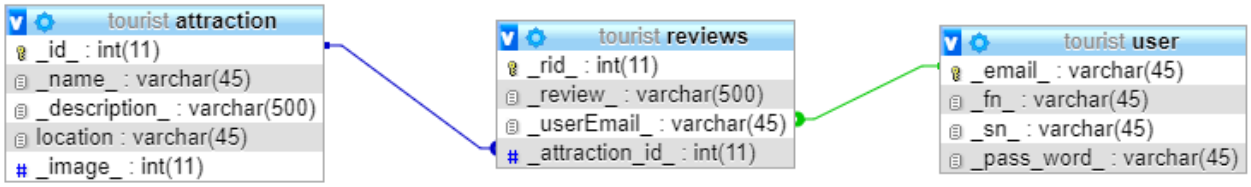


Fig. 8. Database table connections

Fig. 9. Attractions Database

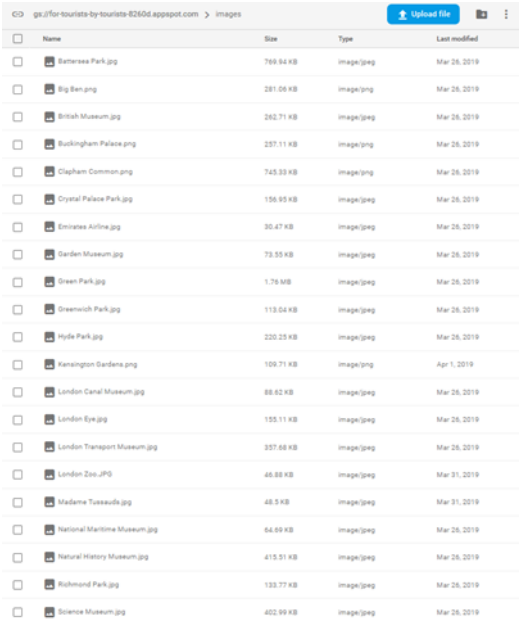


Fig. 10. FireBase Storage

The FOR TOURISTS BY TOURISTS application is using five tables/DataBases (Table 1) which are corresponds to the main objects in the application (attractions, museums, parks, reviews and users tables).

Table 1. Tables in the FireBase DataBase

Table Name:	Purpose:
• **attractions:** • **museums:** • **parks (Parks and Green Places):**	Representing the main attractions in London. Fields: bookingURL, canBook, description, imageURL, location, title.
reviews:	Representing the users' reviews and rating for the corresponding attraction. There is a connection between the reviews table and the attractions table which are both using the attraction title as a key. Fields: attrTitle, rating, review, userName.
users:	The users being registered in the system. Only the users can add attractions, write reviews and give ratings. Fields: email, fName, sName, pass.

The FireBase DataBase has an additional DataBase – for the picture files. In this case they are in jpg format.

CONCLUSIONS

The created application fully achieved the set goals:

1. A practical "client" application has been created, providing access to a central database, attractive in its subject area.
2. New features of the Android API have been developed: ReciclerView widget, Constraint Layout and others. In the new versions of Adroid Studio, they are already built into the environment;
3. An algorithm for access to FireBase (remote non-SQL Database) in the conditions of multithreading has been created and tested, a problem with the synchronization when accessing the database has been solved;
4. Appropriate internal dynamic structures for storing FireBase data have been selected;
5. A pleasant and attractive user interface has been created;
6. Google's authentication mechanism has been reviewed and tested.

REFERENCES

Carolyn, C. & Unger, R. (2012) A Project Guide to Ux Design: For User Experience Designers in the Field or in the Making. (2nd Edition) Berkeley: New Riders.

Griffiths, D. & Griffiths, D. (2017) Head First Android Development. (2nd Edition) Sebastopol: O'Reilly Media.

Liang D. (2012) Introduction to Java Programming Comprehensive Version. (9th Edition) Harlow: Pearson.

Liang D. (2014) Introduction to Java Programming Comprehensive Version. (10th Edition) Boston: Pearson.

Stack Overflow (2019) Available at: https://stackoverflow.com/questions/13814503/reading-a-json-file-in-android/ (Accessed: 07/04/2019).

Kolesnichenko D., Practical programming for Android, Asenevci trade, 2018, ISBN 978-619-7356-55-7 (***Оригинално заглавие:*** Колесниченко Д., Практическо програмиране за Андроид, изд. Асенеци, 2018, трето издание).

Ivanov M., Software application in Android environment, NBU, 2016, ISBN 978-954-535-917-0 (***Оригинално заглавие:*** Иванов М., Софтуерни приложения в среда Андроид, НБУ, София, 2016).

Mitch Tabian, RecyclerView widget - CodingWithMitch.com, 09.2020

FireBase - https://firebase.google.com/, 09.2020