Team 11

QR-Triggered Photo Sharing Application for Museums

Hundesa, Melese Dendena, 165660
ogmele@gmail.com
Yilma, Girma Mamuye 165604
girmamamuye@gmail.com

Abstract

For some historical or technical reasons it is not possible to take pictures in museums. This is because of the fact that in some way the camera reflection might damage the quality of the pictures or Sculpture's, and some museums don't allow taking pictures at all. The idea of this QR-Triggered Photo sharing application project is to create a means for museum visitors to have the experience of enjoying the pictures from the museum by developing an android application to decode a QR-Code encoded with the URL of the images in the museum, bringing low quality version of the picture for image filtering and sharing on social media. While sharing the URL of high quality version of the picture with the selected image filtering applied will be posted on the social media. And to create an interactive dashboard management web application for museum managers, to perform basic operations such as adding pictures, generating QR-Codes and do some basic content management. We believe through this application both museums and visitors will benefit greatly, from the fact that it is easier for visitors to scan a QR-Code than taking the best shots and for the museums, they will have some automated experience for their data in the museums and easily advertise themselves on the social networks.

1. Objectives

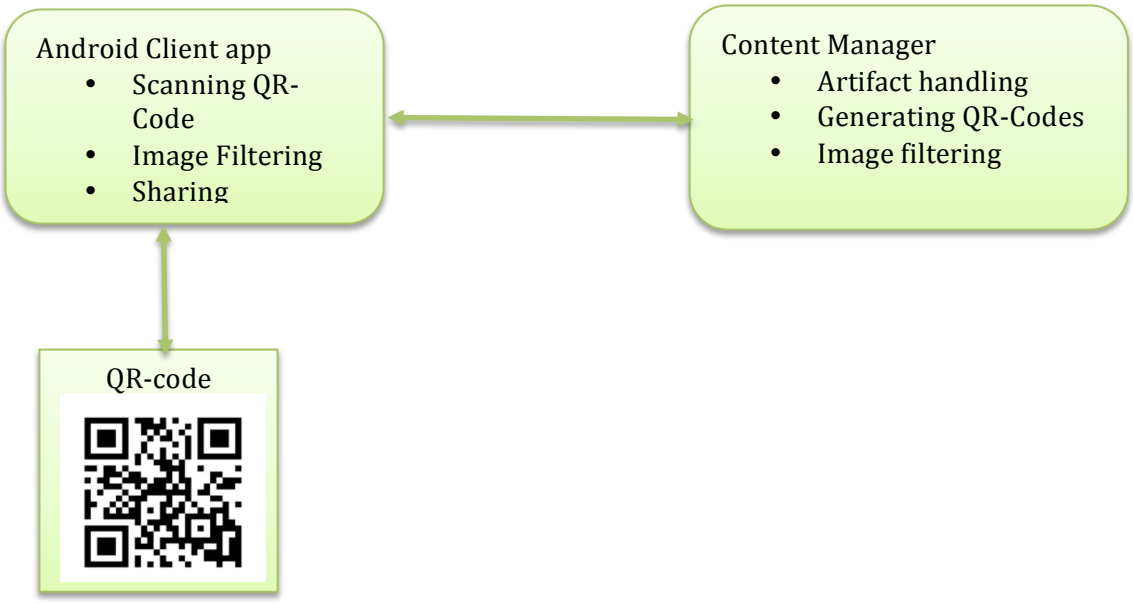
From the Multimedia Networking course point of view, the objective of the project can be explained as; it is a software with multimedia content (images and texts with image processing feature i.e. image compression), a web application for content management and an android client application for users to scan QR-Codes, retrieve an image, do image filtering and share on social media. For bandwidth efficiency we are retrieving a low quality version of the image from the server, which will be generated automatically from the original image (which assumed to be high resolution image) for image filtering. It also enabled us to implement the REST API's for communicating the client and the server side with some UI designs for both the dashboard management system and the client android application. In general the project addresses many of the objectives of the Multimedia Networking course i.e. sending and receiving multimedia content over the network, trying to maintain bandwidth efficiency, creating interactive user interface and allowing us to see the challenges behind the seen in developing multimedia application.

2. Achievements

To describe briefly the main achievements of the project, we are able to develop an android client application with features to scan QR-Codes using the built-in camera, retrieve image through the encoded URL from a server, with multiple image filtering features and sharing capability the URL of the applied techniques for the high quality version of the image on social networks such as Facebook, Twitter, Gmail, Google +, and many others.

On the server side we implemented a content management web applications with features to add pictures, update pictures, generate QR-Codes, print QR-Codes. We also includes different types of view formats for the stored images i.e. gallery view, list view and archive view (to see archive images).

General architecture of the system can be shown as in the following diagram.



For the client part we choose android as a platform because it is one of the dominant and fastest growing mobile operating system and as a mater of fact hundred thousands of android enabled phones are turned on every day, for this reason making this choice would have a significant importance for potential customers. For scanning the QR-Code we used the Google zebra crossing API because it is robust and easy to get the decoded result via intents, which is android feature for sharing messages between activities and among services. It is also, available for different platforms including IOS. The Zebra Crossing library uses the built-in camera to scan QR-Codes and gives back the encoded data as a string. In our case the encoded URL will be reviled and an http request to retrieve the image using the GET command will be sent to the content manager, after getting the low quality image, image filtering can be chosen among different choices by the user and can see the different views on the friendly UI by swiping and clicking among different filtering choices available. And share using the share button to the social media of the user choice.

We believe that we have achieved our objectives at least functionality wise. Our purpose was to build an android application, which will be triggered by QR-Code for image sharing and filtering and a dashboard management system with QR-Code generation

capability and basic content management, and we believe that we have come up with the above listed functionalities. Doing the image filtering on the server side was a bit problematic to test it on real hosting services because of its PHP version requirement and our data size, though we have implemented the code we could not able to find hosting service to test it live. But we make sure that it is working on the local server. And this is because time plays the great role here, and hosting service providers are not flexible enough to test our application (for testing on the requested PHP version only premium services are available though we found some free but data size limitation becomes another problem).

For the Server side we choose PHP as a scripting language and MySQL as a database because of their robustness for web application development even Facebook and many of the existing websites are developed using those technologies. For the skinning of the page we have used a twitter bootstrap, which is a very nice tool to have a good look and feel for webpages, it is a very small code with very many features.
Basically our project has two deliverables, one is the web application for the dashboard management system and an android application for the client side.
To test the web application you can check (http://museummanagement.site11.com/), which is an online-hosted version of the dashboard management system. If anyone wants to launch it on his own server, the following conditions must be meet.

1. PHP version 5.5 or higher is required on the hosting server
2. He should create the database using the query on the file museum.sql
3. Finally the configuration.JSON file for setting up the URL of the new domain, database name, user name and password to be used for accessing the database.
4. Then it is possible to use either file transfer tools such as FIlezila or FTP to put the website online.

For the android application you can install it like any other android applications.
Note: It needs a proper Internet connection to use the application.

3. Conclusions

Some students can extend this project, during the course for next year in the following way. It can be extended in to a real application for museums as a content management with added features such creating albums for a museum artifact, so that multiple images can be viewed for one sculpture, creating a means to rate the number of visit by counting the number of request to a particular museum artifact and many others. One possible proposal for the expansion of the system is it can be made as a service, where museums, can be registered with their credential and get their own copy of the service with some free and paid features. By this it will be possible to create a centralized content management for each registered museums. This would help a lot to develop a robust android application with many features, for example location information's can be added on each QR-Code scanned and put as a travel history, map view of each registered museums with some detail's, visiting plans management also can be incorporated. Finally it will also be possible to develop it for other mobile platforms including IOS, Phonegap to increase number of users.

References

http://www.w3schools.com/
http://developer.android.com/index.html
http://www.php.net/
http://stackoverflow.com/
https://code.google.com/p/zxing/
http://www.genymotion.co/

Annex

Android Client

Main Activity

Number	Attribute Name	Description
1	Scan QR-Code button	Will allow you to move to a new activity to scan a QR-Code generated by our content management web application (other URL's can be decoded also but will not retrieve anything) and will show you an other activity to do image filtering and sharing
2	About button	This button will forward you to another activity which will display some information about the application, what can you do with it etc.

Image Filtering and sharing activity

Number	Attribute Name	Description
1	Swipe buttons for image filtering	Will allow you to select among available filtering techniques to apply on the scanned image
2	Share button	Will pop up with list of sharing applications on social medias you can select one and you will be directed to the selected sharing application with a URL to be shared which basically is an addressed to an image on the server to be displayed on social media's.

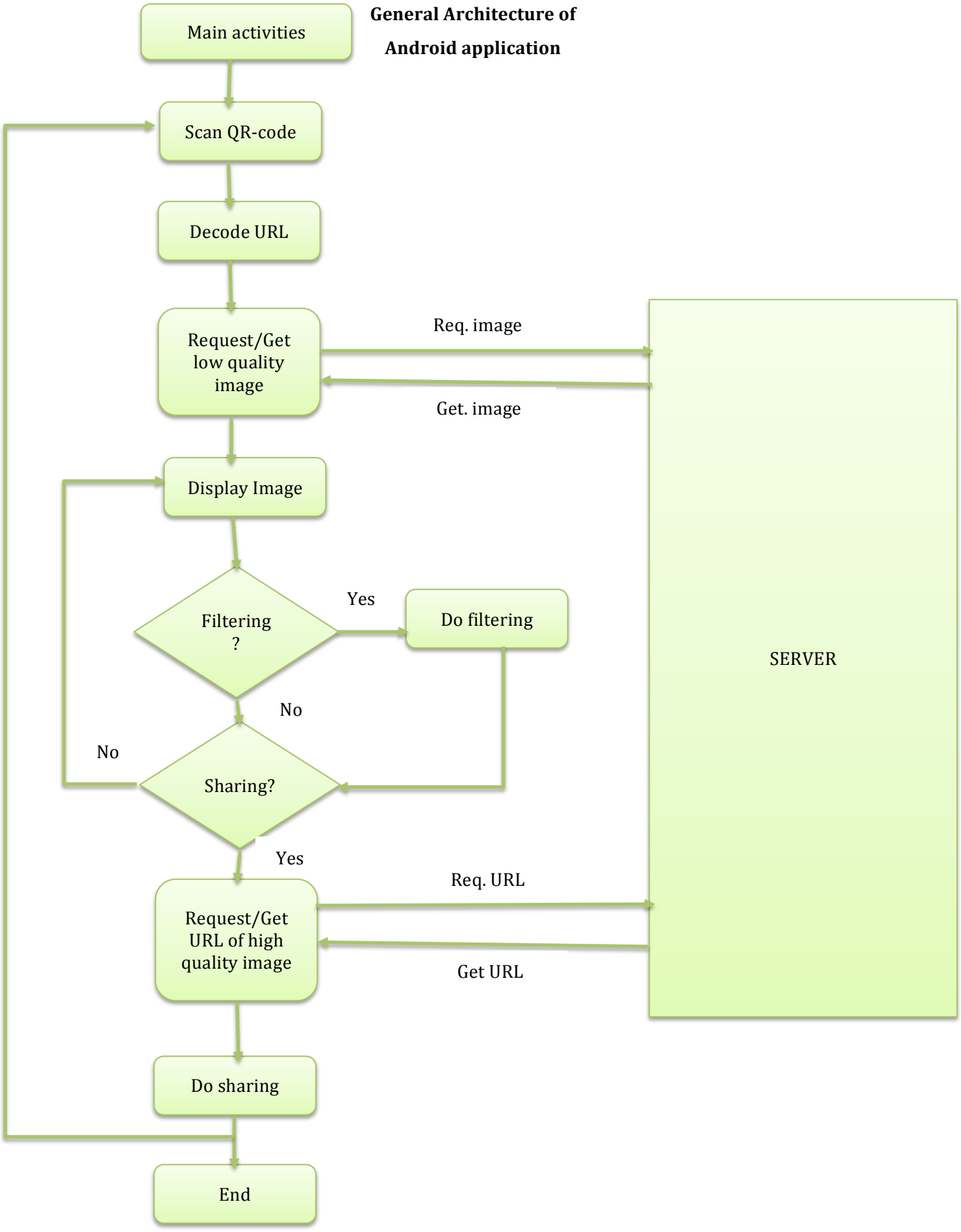
Implementation details

Client side programming language: Native android

Third party QR-Scanning Library used: Google Zebra crossing API

Communication with server: REST client services

Number	Component	Language/Tool	API/Third-party Library
1	Android client	Android version 18	Zxing Library from Google


General Architecture of
Android application
Main activities
Scan QR-code
Decode URL
Request/Get
low quality
image
Req. image
Get. image
Display Image
Filtering
?
Yes
Do filtering
No
No
Sharing?
Yes
Request/Get
URL of high
quality image
Req. URL
Get URL
Do sharing
End
SERVER

Dashboard Management System (DMS)

Database Design

Admin

Number	Attribute Name	Description
1	Full Name	The full name of the person who is going to administer the web site.
2	Email	The email address of administrator that will also serve as a user name. It is also the primary key.
3	Password	Password for the specified Admin
4	Password Reset Code	A random code that will be generated and used in case the user forgets his/her password and asked for reset

Artifacts

Number	Attribute Name	Description
1	Id	The unique identifier of an artifact
2	Name	Name of an artifact
3	High	A path information for the high quality image of the artifact
4	Low	A path information for the low quality image of the artifact
5	Status	And indicator of the status of the artifact. An artifact status can be either *active* or *achieve.*
6	Description	Description for the artifact

Functional Requirements

Admin

Number	Module	Description
1	Admin Registration	Used to register a new admin into the system
2	Admin Account Modification	In case the admin wants to change his password, this will allow him to do so.

3	Login	In order to access the system, the admin should login first using the login functionality.
4	Forgot Password	If the admin forgot his/her password, recovery is possible using the forgot password functionality

Artifacts

Number	Module	Description
1	Adding Artifacts	The system will allow the admin to add new artifacts in to the collection using the add artifacts feature
2	Viewing Artifacts	Once added, artifacts can be viewed as a whole or a particular artifact can be view by using the search functionality
3	Updating Artifacts	Artifacts information can be updated at any given time.
4	Deleting Artifacts	Artifacts information can be deleted at any given time.
5	Archive	Deleting an artifact is a logical operation not a physical. That means, deleted artifacts are achieved not removed from.

Implementation details

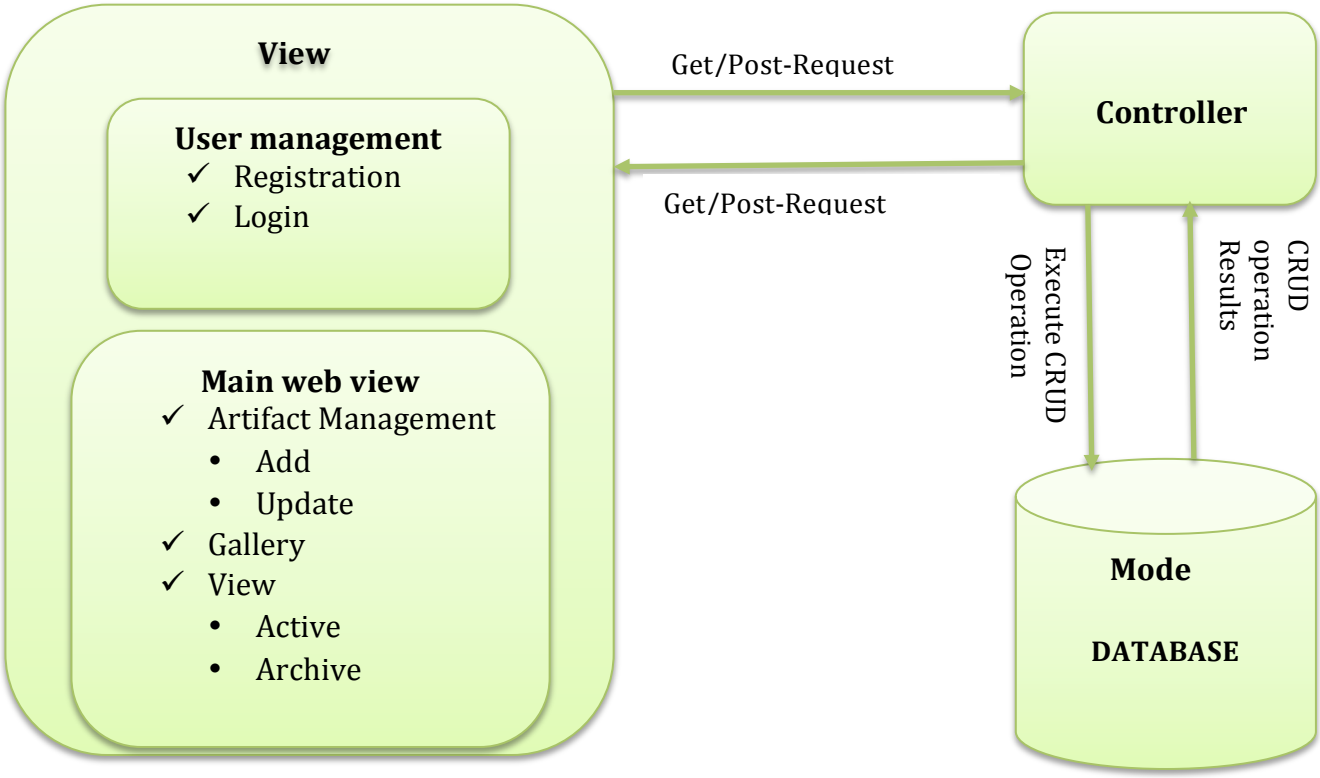
Server side scripting language: PHP 5.5

Third party PHP Library used: QR-Code Generator

GUI: HTML5, CSS3, jQuery, jQuery validation plugin, Bootstrap

Number	Component	Language/Tool	API/Third-party Library
1	Server Side Scripting	PHP 5	QR-CODE generator
2	GUI	HTML5, CSS3, jQuery	Bootstrap, jQuery validation plugin
3	Database	MySQL	None

High-level architecture of the Dashboard Management System (Model View Controller)



Description on some features

Image-Compression: is done using the PHP's built in GD [1] library

The method used is:

bool **imagejpeg** (resource $image [, string $filename [, int $quality]])

imagejpeg() creates a JPEG file from the given image.

Parameters ¶

Image

An image resource, returned by one of the image creation functions, such as imagecreatetruecolor().

Filename

The path to save the file If not set or **NULL**, the raw image stream will be outputted directly.

To skip this argument in order to provide the quality parameter, use **NULL**.

---

[1] http://php.net/manual/en/function.imagejpeg.php

[2] http://www.phpclasses.org/package/6399-PHP-Generate-QR-Code-images-

Quality

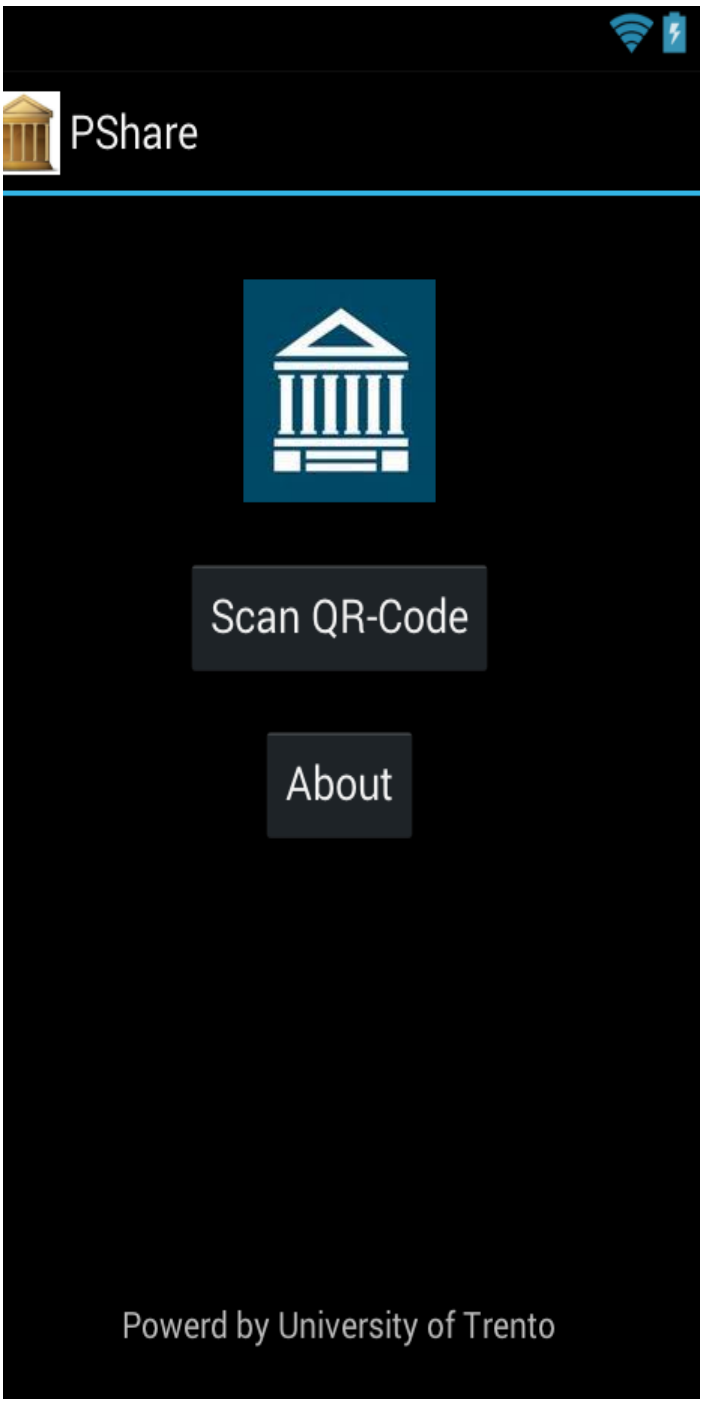
> Quality is optional, and ranges from 0 (worst quality, smaller file) to 9 (best quality, biggest file). The default is the default IJG quality value (about 7).

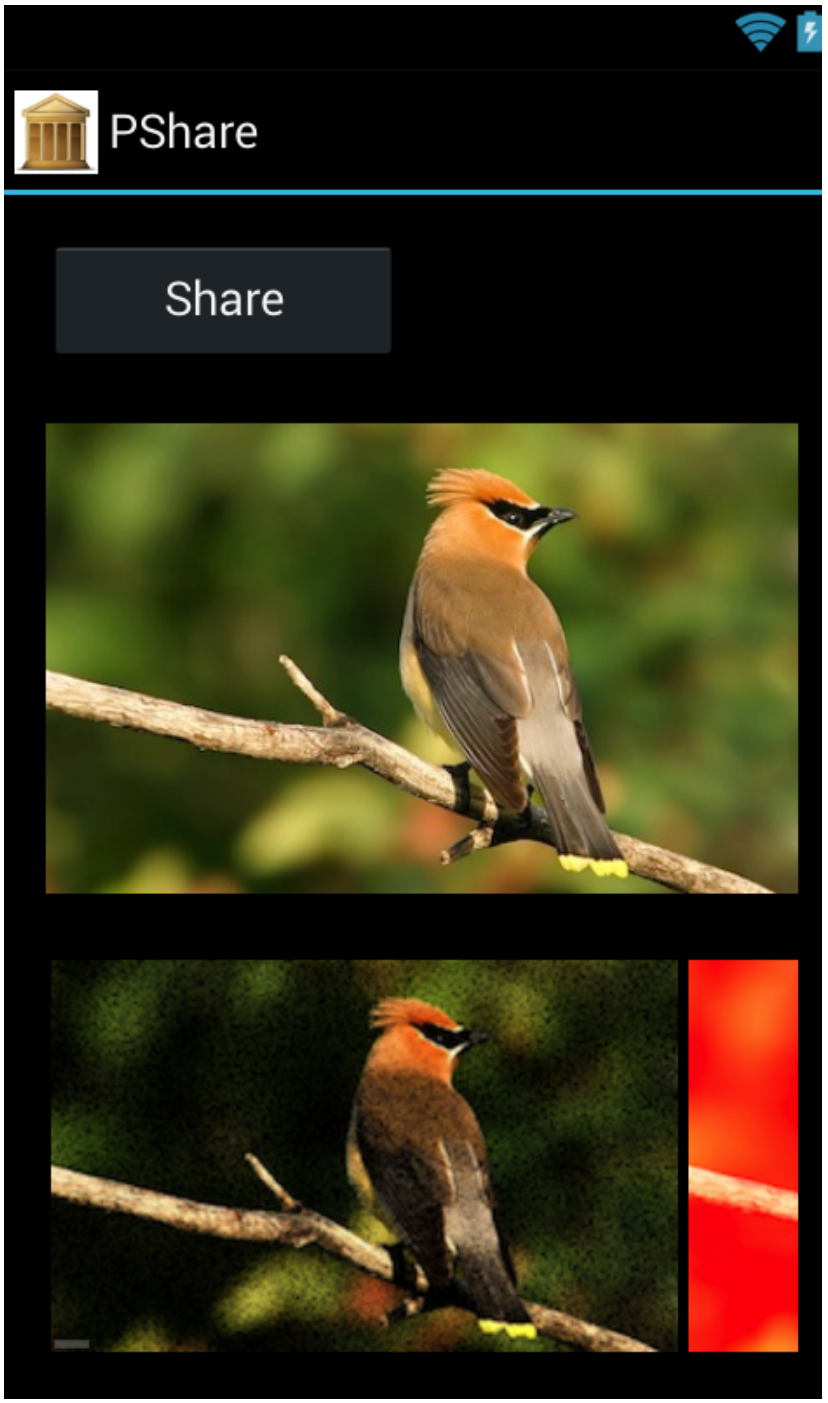
Return Values ¶

Returns **TRUE** on success or **FALSE** on failure.

QR-Code generation: is done using third party [2] QR-Code generator

[2] http://www.phpclasses.org/package/6399-PHP-Generate-QR-Code-images-using-Google-Chart-API.html





Some screen shots from the applications