How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

Description
Intended User
Features
User Interface Mocks
- Main screen
- Menu unfolded
- QR scanning screen
- Create QR code
- Edit QR code
- Scanned QR code
- QR code details

Key Considerations
- How will your app handle data persistence?
- Describe any corner cases in the UX.
- Describe any libraries you'll be using and share your reasoning for including them.
- Describe how you will implement Google Play Services.

Next Steps: Required Tasks
- Task 1: Project Setup
- Task 2: Implement UI for Each Activity and Fragment
- Task 3: Generation of QR Codes
- Task 3: Sharing QR Code
- Task 4: Create local database and content provider
- Task 5: Implement server backend
- Task 6: Implement Google Analytics interface

GitHub Username: bongo1986

QRDB

Description

Lets you share whatever whatever information you want with other people via QR code.

Intended User

This app is for everybody willing to share a piece of information with people in nearby. For example you could stick a label with QR code generated by the app on your office door. People could later scan it and learn more about you and your company or you could stick a label with QR code generated by the app on your motorbike and share data about it with other motorcycle-lovers.

Features

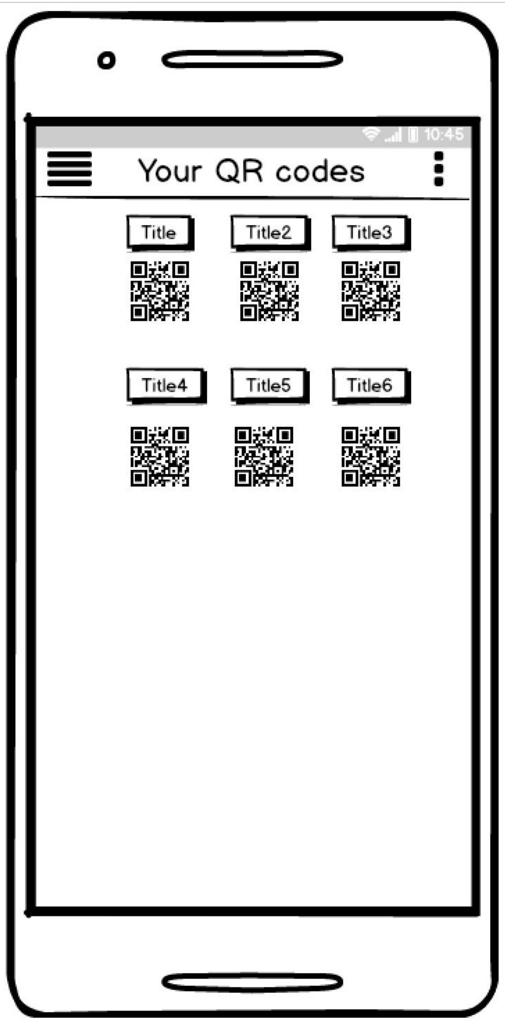
The app will support following features:

- Generating custom QR codes and assigning text information to QR codes.
- Sharing QR codes via Email or MMS so they can be later printed.
- Scanning QR codes and displaying text informations assigned to them.
- App will provide a widget showing how many times user's QR codes have been scanned by others.

User Interface Mocks

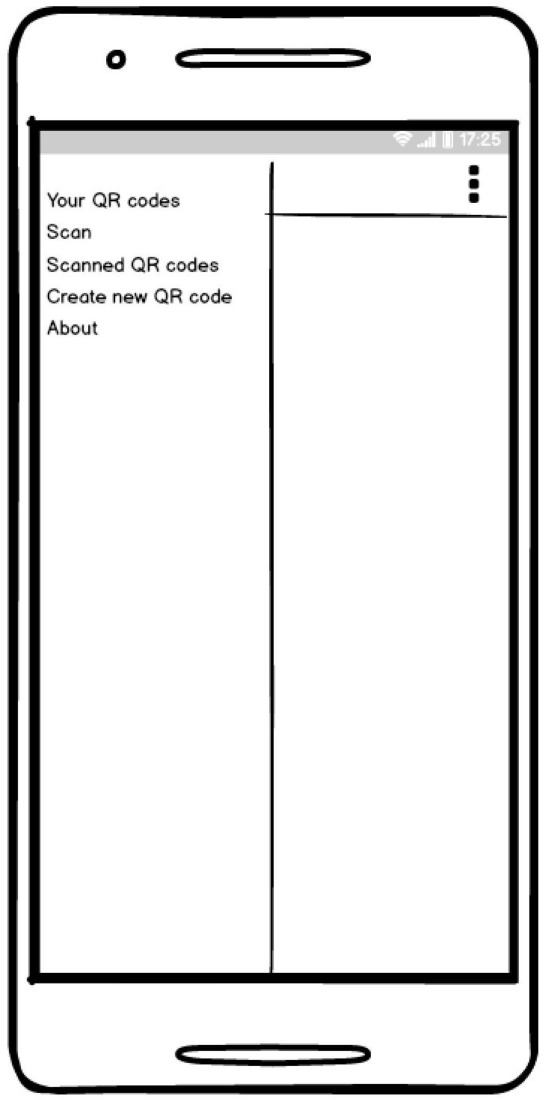
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Main screen



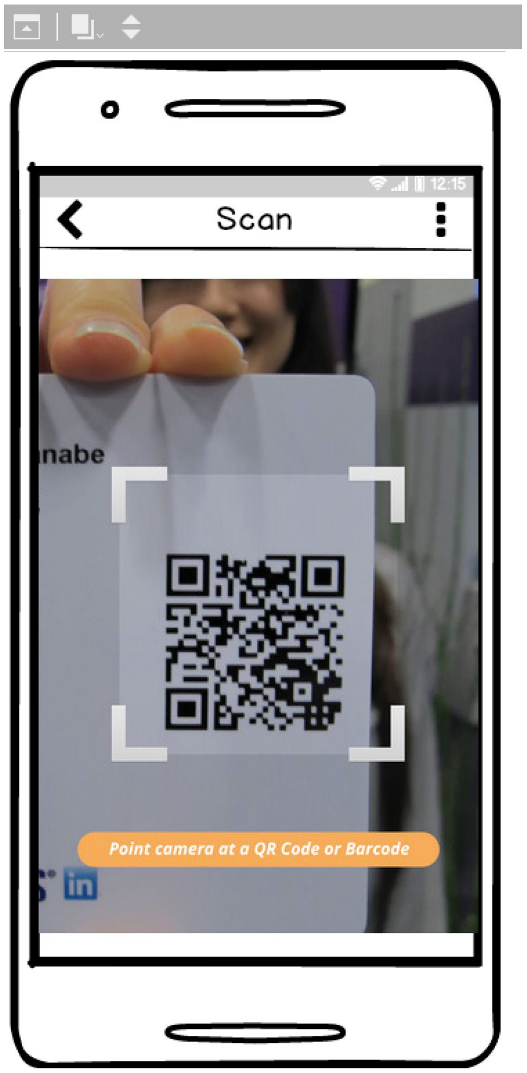
This is the first screen displayed after starting the app. It shows a list of QR codes generated for the user together with their titles.

Menu unfolded



The app will provide side menu where user can choose one of following options: “Your QR codes”, “Scan”, “Scanned QR codes”, “Create new QR code”, “About”.

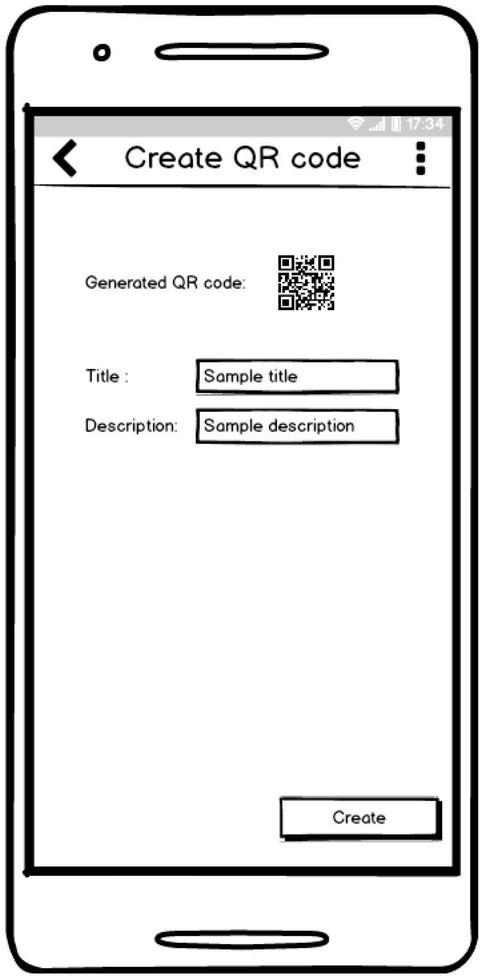
QR scanning screen



The user can scan QR codes on “Scan” screen. If the scanning is successful QR code and its details are added to local database and user is redirected to “QR code details” screen

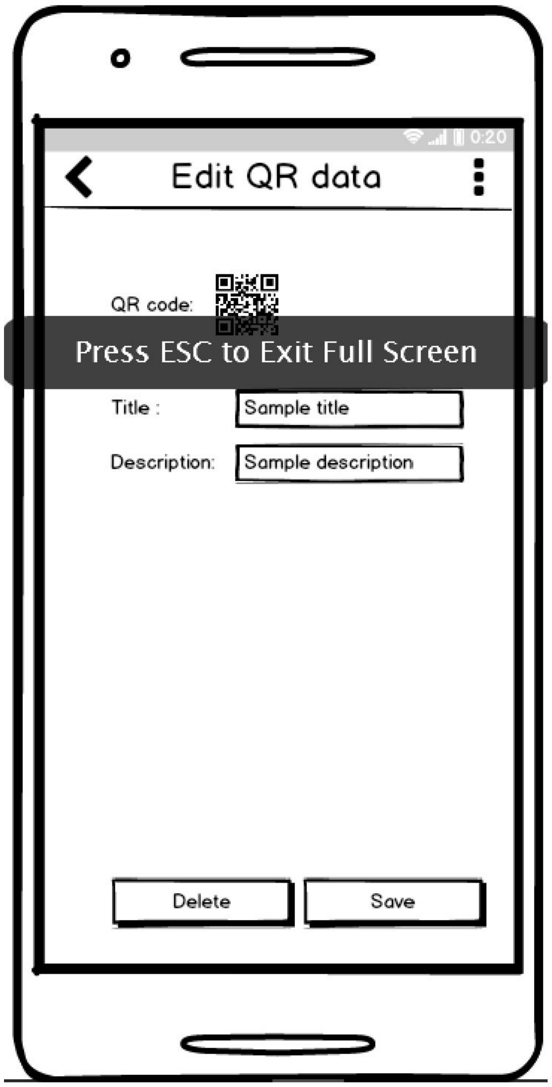
Create QR code

The user can assign a piece of information with a QR code on this screen. QR code is generated with zxing library and represents a GUID. All input fields on the screen are required. Title field can not be longer than 50 characters. Description field can not be longer than 500 characters.The record will be created in the local database as well as on the Google Cloud backend.



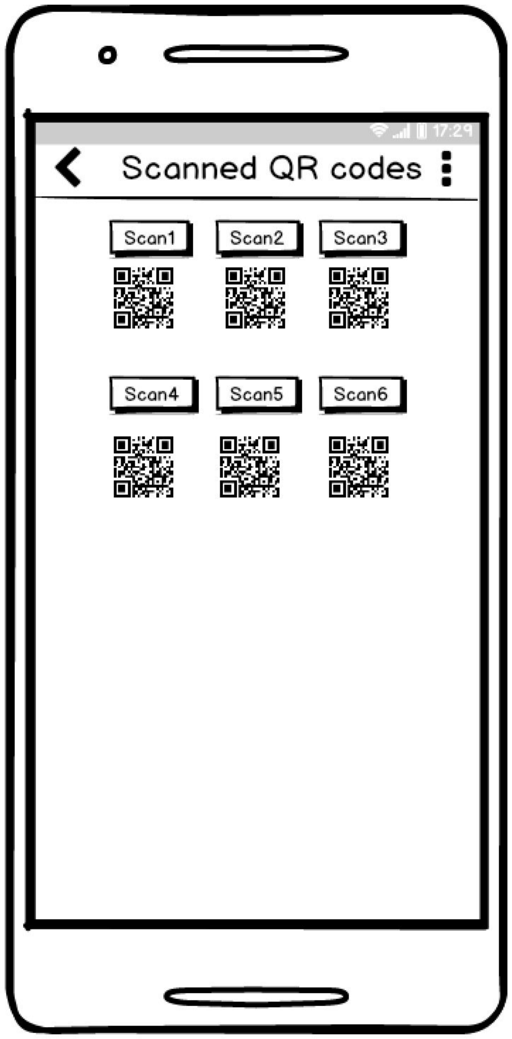
Edit QR code

User can edit texts assigned to QR code. All fields are required. Title field can not be longer than 50 characters. Description field can not be longer than 500 characters. The user can also delete QR code together with the data assigned to it. The record will be deleted from a local database as well as from Google Cloud backend



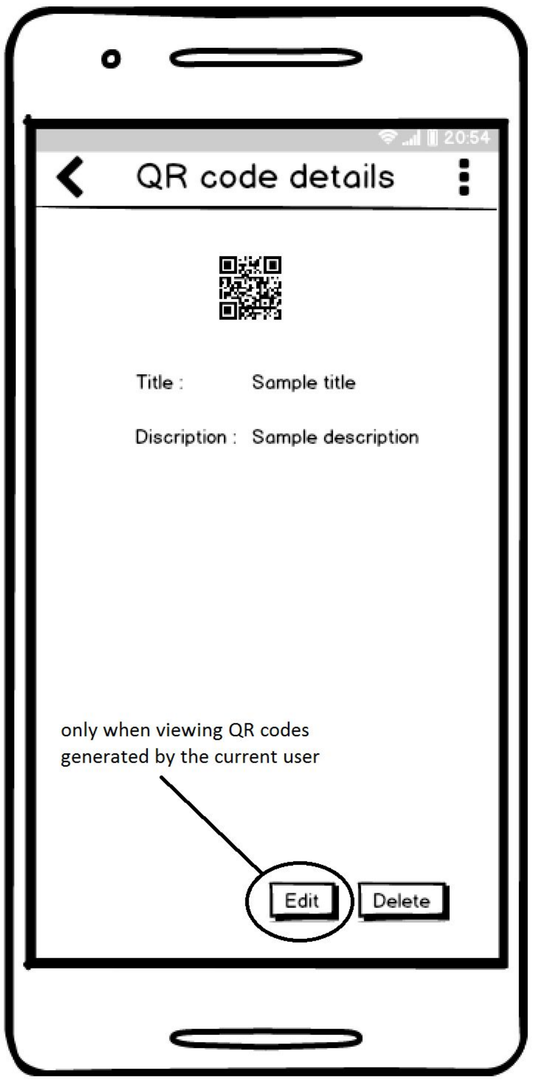
Scanned QR code

This screen displays QR codes that have been scanned on the current device. The data displayed is stored in database.



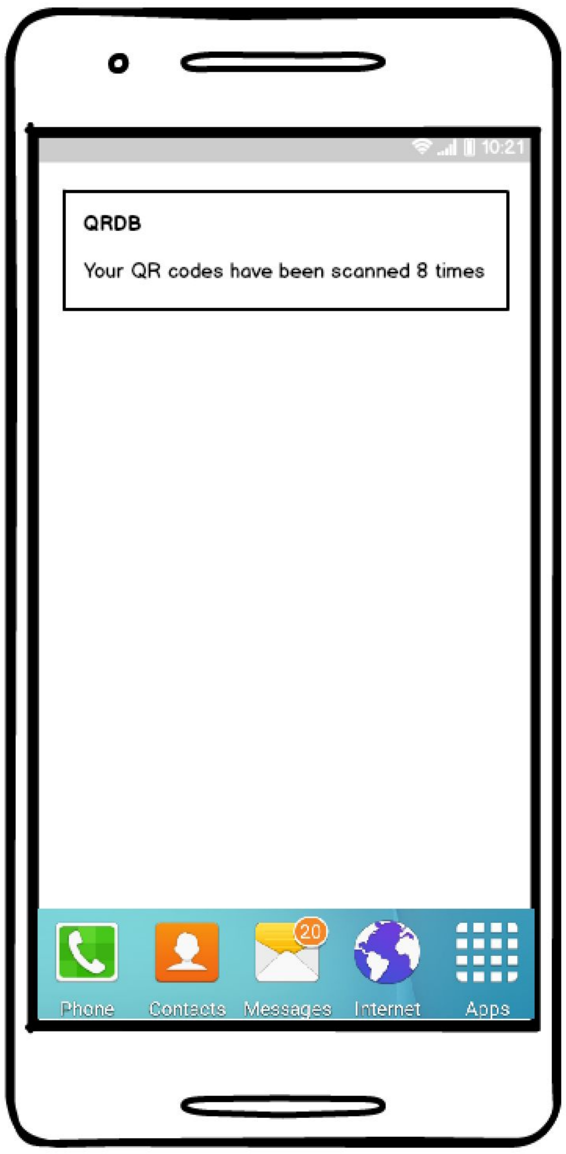
QR code details

QR code details screen display the title and description together with an image of QR code, When a user deletes a scanned QR code only the record from a local database is removed.


20:54
QR code details
Title : Sample title
Discription : Sample description
only when viewing QR codes generated by the current user
Edit
Delete

QRDB widget

Widget will show how many times users QR codes have been scanned by other users. The scan count will be stored locally and will be periodically updated by a sync adapter.



QRDB

Your QR codes have been scanned 8 times

Key Considerations

How will your app handle data persistence?

The app will use a custom content provider retrieving data. QR codes together with data will be stored on Google Cloud. There will be a local SQL database on the device storing data about scanned QR codes and user’s QR codes. Whenever user deletes his own QR code the record will be deleted from a local database and from Google Cloud backed.

Describe any corner cases in the UX.

User starts the app on the “Main screen” from which he can navigate to “QR code details” screen by tapping on a chosen QR code. User can whole time navigate between different screens in the application by using side menu.

Describe any libraries you’ll be using and share your reasoning for including them.

https://github.com/zxing/zxing : this library will be used to support generation and scanning of QR codes.

http://square.github.io/picasso/: this library will be used to load images of QR codes.

https://github.com/ReactiveX/RxAndroid: this library will be used to handle concurency and threading issues(if needed)

The list is not final

Describe how you will implement Google Play Services.

Google analytics is going to be used for monitoring user activities on different screens. Google Could is going to be used as a server backend to store QR codes database.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

1. Create an Android project
2. Include all the necessary libraries

Task 2: Implement UI for Each Activity and Fragment

1. Build UI for Main screen (think about tablet design).
2. Build UI for side menu.
3. Build UI for QR scanning screen.
4. Build UI for Create QR code.
5. Build UI for Edit QR code.
6. Build UI for Scanned QR code.
7. Build UI for QR code details (think about tablet design).

Task 3: Generation of QR Codes

1. Find out how to use zxing library to generate and scan QR codes.
2. Find out how to store unique GUID in QR code. The GUID will be later used as an identifier for information provided by the user.
3. Implement QR code generation functionality.
4. Implement QR code scanning functionality.

Task 3: Sharing QR Code

1. Implement sharing QR code picture via Email.
2. Implement sharing QR code picture via MMS.

Task 4: Create local database and content provider

1. Create local database.
2. Create content provider to interface local database.
3. Use content provider to save QR codes generated for user
4. Use content provider to read QR codes stored in database.
5. Implement loader to show the collection of QR codes on the screen
6. Implement CRUD functionality for QR codes.

Task 5: Implement server backend

1. Find out how to use Google Cloud to store QR codes together with information provided by the user
2. Extend content provider to support Google Cloud web service.

Task 6: Implement Google Analytics interface

1. Investigate how to use Google Analytics to track user activity on different screens.
2. Implement Google Analytics functionality.

Task 7: Implement QRDB widget

1. Extend local db to support QR code scanning count.
2. Implement SyncAdapter to synchronize local scanning count with server backend.
3. Implement widget UI.

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"