Software Requirements Specification

for

QR Code Generator

Version 1.0 approved

Prepared by: Jacob Heebner, Ben Zoiss, Marcellus Hunt, Andrew Chittick, and Steven Downing

IUPUI

10/11/2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
All	10/18/19	Initial Report	1.0

1. Introduction

1.1 Purpose

To create a QR generator. Version 1.0.

Takes user input and generates a QR code that is available to download as a PNG.

“A QR code (short for "quick response" code) is a type of barcode that contains a matrix of dots. It can be scanned using a QR scanner or a smartphone with built-in camera. Once scanned, software on the device converts the dots within the code into numbers or a string of characters. For example, scanning a QR code with your phone might open a URL in your phone's web browser.” (https://techterms.com/definition/qr_code)

1.2 Document Conventions

TBD

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

1.3 Intended Audience and Reading Suggestions

This project is intended for any user who wishes to condense their information into QR codes.

1.4 Product Scope

The QR generator will support Numeric and Alphanumeric modes and character count versions 1-40 with error correction levels L, M, Q, and H. The QR generator will be a desktop application.

1.5 References

QR Code algorithm found at:
https://www.thonky.com/qr-code-tutorial/
LodePNG encoding/decoding software:
https://github.com/lvandeve/lodepng

2. Overall Description

2.1 Product Perspective

The QR generator is a new self-contained product.

2.2 Product Functions

The product will output a QR code for the user. Users will input their text data on the first screen and the QR software will do the algorithms and conversions and output them to a second screen where the user can download a scannable QR code.

2.3 User Classes and Characteristics

This is a general use software available to the general public, so there are currently no plans to add different classes of users. Each user has access to the same software, same limitations on convertible data, and can share their codes with whoever they want.

2.4 Operating Environment

The software will be created using C++ and the QT framework. Users will be able to download the application and run it on their desktop PC, Mac, or Linux.

2.5 Design and Implementation Constraints

One notable constraint that will require attention to design is the size of the data the user wishes to convert. Given the regular QR code size can hold 406 bytes of information, this could limit the size of data the user includes. The project will also be limited to designing codes that can be read by QR scanners. Security concerns and policies don't readily apply to this as it is a software meant to be used by anybody as a way to store information in a quickly readable image. The Qt framework library will be used for the graphical interface. LodePNG will be used to convert a vector into a viewable image.

2.6 User Documentation

There is currently no planned documentation to be included, but it can be possible to include a readme file to explain the process and menu options the user will encounter or a help button to provide information on the current options.

2.7 Assumptions and Dependencies

This software relies on access to the qt library and LodePNG (C++ PNG encoder/decoder) in order to code this software in C++. It is assumed that choosing to code this project in the C++ language will not limit the user in entering and encoding data outside of the constraints expected for the QR code itself.

3. External Interface Requirements

3.1 User Interfaces

The first screen of the program will have a text box area for users to type in data or links to convert into QR code format. It will also have a submit button that runs the program to convert all the data. Error messages will be displayed for incorrect data, whether it be format or incompatibility.

The second page of the program will have a download button for users to download their QR code in a PNG format. It will also have a display for a preview of the QR image.

3.2 Hardware Interfaces

The software will be written in C++ and (after a one-time compilation) an executable file will be used to run the program. The user will use the keyboard and mouse to interact with the program. The program will also interact with the users file system to save the QR Code PNG if desired.

3.3 Software Interfaces

Our software will interact with PNG encoding software. LodePNG version 20190914 is a C++ file that converts a C++ vector into a PNG image. After creating a C++ vector with the relevant information (vector of unsigned chars, RGBA layout found here https://github.com/andrewChittick/makePNG/blob/master/vectorLayout.jpg) to make a QR code, we will pass a pointer to this vector along with size information to LodePNG which creates a PNG that it can store to disk (current prototype use) or other options such as displaying it. A prototype on how our software interacts with LodePNG version 20190914 can be found here https://github.com/andrewChittick/makePNG.

3.4 Communications Interfaces

N/A

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 Dynamic Image Creation

4.1.1 Description and Priority

Based on user input and the QR code generation algorithm we will create a dynamic QR code. The system cost can range from 1 to 7 based on how long the alphanumeric string is.

4.1.2 Stimulus/Response Sequences

The user will enter their alphanumeric string, a primary color, and an image to watermark in the background. They will press the button to process the input and there will be a new button that they can press to download the QR code.

4.1.3 Functional Requirements

The user will need to be using an operating system with a GUI. They will also need to have the g++ compiler along with the QT framework.

REQ-1:	g++ compiler
REQ-2:	QT framework
REQ-3:	GUI operating system

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The product must perform on as many different systems as we can. Using different methods to compute data would be important depending on what kind of device the current user is running (i.e. multithreading when on devices that support it).

5.2 Safety Requirements

N/A

5.3 Security Requirements

N/A

5.4 Software Quality Attributes

The main goal is to have the software be an easy to use product to generate QR codes. Availability is also important, as we want to have our software be able to run on as many platforms as possible. Adaptability and maintainability will also be important aspects to implement so that new features may be added, such as new sets of data types or input methods.

5.5 Business Rules

Anyone in their business can condense the information into a QR code.

6. Other Requirements

There will be an SQL database to store the tables for the conversions in error correction codewords.

Appendix A: Glossary

N/A

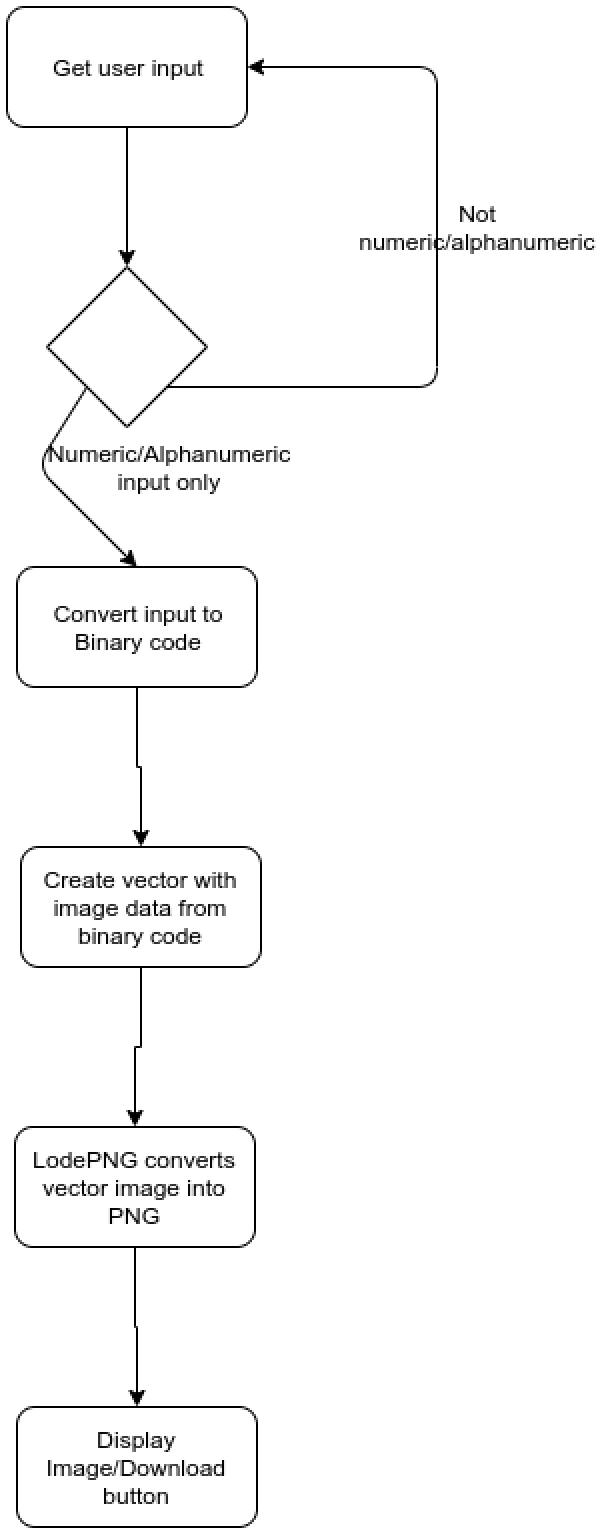
Appendix B: Analysis Models

Shown below.

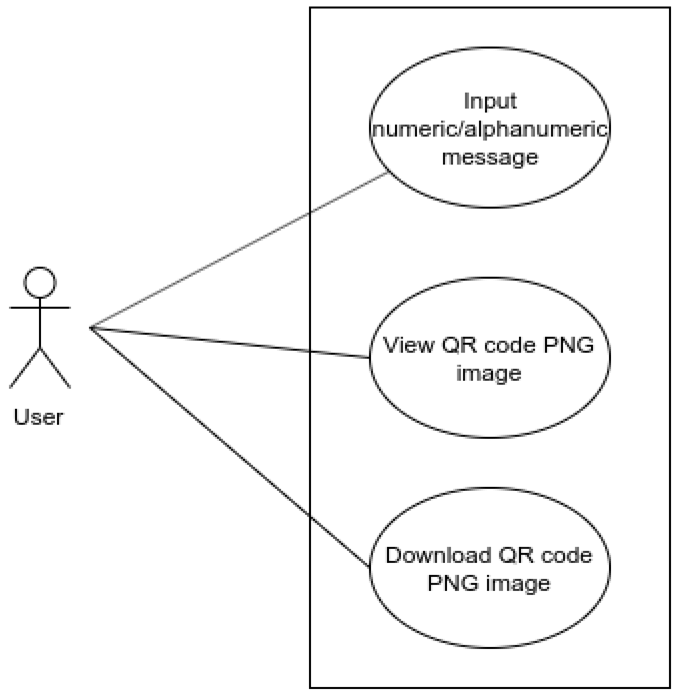
Appendix C: To Be Determined List

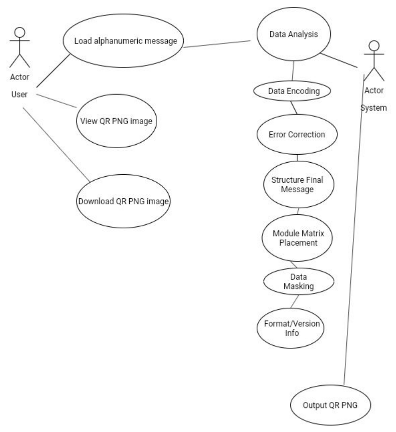
<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

Activity Diagram


Get user input
Not numeric/alphanumeric
Numeric/Alphanumeric input only
Convert input to Binary code
Create vector with image data from binary code
LodePNG converts vector image into PNG
Display Image/Download button

Use Case Diagram


Input numeric/alphanumeric message
User
View QR code PNG image
Download QR code PNG image


Actor
User
Load alphanumeric message
View QR PNG image
Download QR PNG image
Data Analysis
Data Encoding
Error Correction
Structure Final Message
Module Matrix Placement
Data Masking
Format/Version Info
Output QR PNG
Actor
System