Software Requirements Specification

for

Resistor Value Recognizer

**Version 1.0 approved**

**Prepared by Husnal Kaur, Brendan Woo**

**Humber College Computer Engineering Technology**

**2019-10-02**

**Table of Contents**

[**1.**](#_heading=h.1fob9te) **Introduction 1**

[1.1](#_heading=h.3znysh7) Purpose 1

[1.2](#_heading=h.2et92p0) Product Scope 1

[1.3](#_heading=h.tyjcwt) Intended Audience and Reading Suggestions 1

[1.4](#_heading=h.3dy6vkm) References 1

[**2.**](#_heading=h.1t3h5sf) **Overall Description 1**

[2.1](#_heading=h.4d34og8) Product Perspective 1

[2.2](#_heading=h.2s8eyo1) Product Functions 2

[2.3](#_heading=h.17dp8vu) User Classes and Characteristics 2

[2.4](#_heading=h.3rdcrjn) Operating Environment 2

[2.5](#_heading=h.26in1rg) Design and Implementation Constraints 2

[2.6](#_heading=h.lnxbz9) User Documentation 2

[2.7](#_heading=h.35nkun2) Assumptions and Dependencies 2

[**3.**](#_heading=h.1ksv4uv) **External Interface Requirements 3**

[3.1](#_heading=h.44sinio) User Interfaces 3

[3.2](#_heading=h.2jxsxqh) Hardware Interfaces 3

[3.3](#_heading=h.z337ya) Communications Interfaces 3

[**4.**](#_heading=h.3j2qqm3) **System Features 3**

[4.1](#_heading=h.1y810tw) System Feature 1 3

[4.2](#_heading=h.4i7ojhp) System Feature 2 (and so on) 4

[**5.**](#_heading=h.2xcytpi) **Other Nonfunctional Requirements 4**

[5.1](#_heading=h.1ci93xb) Performance Requirements 4

[5.2](#_heading=h.3whwml4) Safety Requirements 4

[5.3](#_heading=h.2bn6wsx) Security Requirements 4

[5.4](#_heading=h.qsh70q) Software Quality Attributes 4

[5.5](#_heading=h.3as4poj) Business Rules 5

[**6.**](#_heading=h.1pxezwc) **Other Requirements 5**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Woo, Kaur | 2019-10-02 | Initial report | 1.0 |

# Introduction

## Purpose

The project is a Resistor Value Recognizer (RVR) app. Its main functionality is to connect to a device which scans and displays the colour coded value of a resistor. Due to the large volume of students that use the parts crib, significant amounts of unsorted components often accumulate from returned components. High traffic makes it difficult to sort these components quickly, and it is often time consuming to figure out each and every value of the resistors. Moreover, it can be hard to identify resistor values correctly in a fast-paced environment. Current revision number is 1.0.

## Product Scope

The app communicates with a database which records usage of the Resistor Value Recognizer hardware platform for each user. It also provides an interface with which to communicate with the companion device and a GUI to view resistor values. Both live scans and the history of previous scans can be viewed via the app, making for easy organizing of resistors. The goal is to provide a user-friendly means of controlling the device and accessing their data hosted on the server.

## Intended Audience and Reading Suggestions

This document is targeted towards the following types of audience:

* Humber College Parts Crib Employees
* Students from various Schools, Colleges and Universities
* Professionals working in a fast paced environment
* Developers looking to develop applications interfacing with the server

For the best understanding of this document, the readers are suggested to become familiar with concepts like server, database, GUI(Graphical User Interface) and information communication.

## References

These sources were used in the ideation of this project, and can be used for reference.

* Amazon. (2019, September 05). Amazon Go. Retrieved from Amazon.com: <https://www.amazon.com/b?node=16008589011>
* Cruz, J., Dimaala, M., Francisco, L., Franco, E., & Bandala, A. (2013). Object recognition and detection by shape and color pattern recognition utilizing Artificial Neural Networks. 2013 International Conference of Information and Communication Technology (ICoICT).

# Overall Description

## Product Perspective

This project is a Capstone Project for Humber College’s Computer Engineering Technology program. As a result, this is a standalone project consisting of both the hardware and software of a companion device, a web server, and the Android app. The app and companion device will communicate via the web server, which stores relevant user data as well as data used to control the companion device from the app.

Data flow:

****

## Product Functions

The app provides the following core functionalities, explained further in Section 3

* Communications with the database
* History viewing
* Resistor Color Code Lookup table
* User account management

## User Classes and Characteristics

This application and the companion device are mainly targeted towards three different types of users:

1. Humber College Parts Crib: The parts crib has, at any given time, high amount of various components which are to be sorted in different spaces. This device and app will help them quickly identify resistor values without going through the process of calculating them manually.
2. Students (Schools/Colleges/University): Students who are unfamiliar with resistor color codes and how to calculate the values of resistors can easily use the app and the device to find the value without having to look up online.
3. Professionals working in a fast paced environment: This app will bring ease of resistor identification to professionals who use multitude of resistors on a daily basis. It will be easier and faster for them to get the values and store them in the history of the application than either calculating the value every time or putting the resistor inside the physical device again.

## Operating Environment

The software will operate as an app on an Android phone running Android 7 (Nougat), API 24. The targeted minimum API of the app is 19 (Kitkat) to ensure compatibility. It must run alongside other applications which may be making use of the same hardware resources, including the camera and Wi-Fi.

## Design and Implementation Constraints

* The app and its features can only be accessed by using an account created on the app. Without creating an account, the user can’t use the history feature or view the initial scanning/QR screen to connect to the device via the app.
* The physical device and the app can only work in the vicinity of Wi-Fi connection, since all the data flow and calculation happens on the server and a network connection is necessary for communication.
* The app will only work in conjugation to the hardware device, without the device the app has no recognition power of its own. A resistor colour code lookup table will be included in the app to provide some functionality to the user without the additional hardware.

## User Documentation

All documentation is available on the following Github repositories:

* <https://github.com/BW25/Resistor-value-recognizer>
* <https://github.com/HusnalK/Resistor-Value-Recoganizer-RVR>

## Assumptions and Dependencies

The operating environment must have access to power and Wi-Fi for the companion device, and a Wi-Fi or cellular data connection for the phone to communicate with the database. The database will be a Google Firebase server, which must be online for the app to connect to it.

# External Interface Requirements

## User Interfaces

The app’s user interface is the GUI displayed on the Android device. A menu bar will be provided at the bottom of the screen for navigation through app activities. The activities accessed via the menu bar include:

* Scan activity: viewing live scans from the device
* History activity: viewing past scans
* Lookup activity: access a resistor code lookup table
* Settings activity: Change app settings, user account info, or connect to a companion device

The app opens with user login and device connection (if necessary) and then the user is sent to the Scan screen, from which they can navigate to other activities via the menu bar.

The app GUI mockup can be found with the following link:

<https://github.com/BW25/Resistor-value-recognizer/blob/master/documentation/Software%20(Ceng%20319)/Ceng%20319%20App%20mockup%20BWoo%20HKaur.pdf>

## Hardware Interfaces

The app is designed for Android devices with a GUI created for Android phone screens. The minimum API targeted is intended to be API 19, Kitkat, to ensure compatibility with 95% of Android devices, as of the date of this document’s creation. The phone’s Wi-Fi capabilities will be used to connect to a Google Firebase server to access data from the hardware. The phone’s camera will also be used to scan QR codes that will be used to identify to the app which device they should connect to.

## Communications Interfaces

The web server will serve as the communications interface between the app and the companion device. User data will be communicated for user login and account management. Details of resistor scans, including resistor values and scan timestamps are expected to be the most common data received by the app. The app will also send data for the purpose of controlling the device, specifically, logging the user’s usage of the device and triggering the companion device to scan.

# System Features

## Communications with the database

4.1.1 Description and Priority

The app must be capable of connecting and communicating with the web server for sharing data with the companion device, which also sends data to the web server.

This requirement is the highest priority (9), as it is a prerequisite for requirements 2 (User account management) and 3 (History viewing).

4.1.2 Stimulus/Response Sequences

Inputs will come from the user, who will scan a QR code on the device. The QR code stores the name of the device, which identifies it on the database, allowing the app to communicate with it. The necessary data to connect to the database will be stored internally by the app. Outputs will appear on the GUI, indicating the status of the device.

4.1.3 Functional Requirements

REQ-1: App connects with database

The app must initiate communication with the web server and connect to access the data in the database. The app’s control of the database must be set so the app may only edit the user’s own information under certain conditions, and only manage their own history.

REQ-2: App updates device command table to control the device

The app must trigger the device to take a picture of a resistor and extract the colour code, then log it to the database

## User account management

4.1.1 Description and Priority

The user must be able to create a new account or log into an existing one. They must also be able to modify their account via the Settings activity. This requirement is the second highest priority (8) as it has the prerequisite of Requirement 1 (Communications with database), and is a prerequisite for Requirement 3 (History viewing).

4.1.2 Stimulus/Response Sequences

The user will input their username and password from the login/signup activity, which will be used to control the user’s access to the database. Only when the user is logged in will they be able to progress to the other activities.

4.1.3 Functional Requirements

REQ-1: Sign up

A new user must be able to create a new account to access the database and store their data within it.

REQ-2: Log in

A user with a pre-existing account must be able to log in to access their data.

REQ-3: Update user information

A user should be able to modify their account information, including changing their email, password, and deleting their account

## History viewing

4.1.1 Description and Priority

The app must be capable of viewing data of resistor scans corresponding to the logged in user and displaying it in a GUI for the user. This has the third highest priority (7), as it is one of the core functionalities of the app, but has the prerequisite of Feature 1 (Communications with companion device), and Feature 2 (User account management).

4.1.2 Stimulus/Response Sequences

Inputs will include the user’s id, obtained when the user logs in. This allows the app to use the database connection to access the resistor scans belonging to that user.

4.1.3 Functional Requirements

REQ-1: Access user’s history

The database connection will be used to read from the table storing the history of the logged in user

REQ-2: Display history to GUI

The history will be displayed to the History activity, with the most recent scan appearing in both Scan activity and History activity.

REQ-3: Delete user’s history

Users should have the option of deleting their history to give the user control over their data and free space in the database

## Resistor colour code lookup table

4.1.1 Description and Priority

The user must be able to access a resistor code lookup table used as a learning tool, or in the event a connection to the database is not possible. This requirement has a lower priority (5), as it is not required for the core functionality of the app and serves as a backup in the event other features are not available due to lack of wireless connection.

4.1.2 Stimulus/Response Sequences

The only input required is the user selecting the Colour Code activity, which displays the colour code lookup table to the GUI

4.1.3 Functional Requirements

REQ-1: Colour code lookup activity

The colour code activity must be created and accessible through the menu bar

REQ-2: Lookup table

The lookup table must be created and stored in the app’s resources so it can be displayed to the GUI

# Other Nonfunctional Requirements

## Performance Requirements

The project heavily depends on the database connection to and from the device and the app. High speed wifi and/or mobile data connections are needed for optimum performance. The information sent to the database for establishing the connection between the app and the device via the QR code scanner should also be accurate for the functioning of the system. The system must be capable of recognizing resistors quickly to have utility and increased usability compared to the user figuring out the resistor codes themselves.

## Safety Requirements

The app must not make unnecessary use of the phone’s resources and take care not to have data leaks that could affect other functionalities of the phone. Upon uninstalling the app, all relevant data should be removed.

## Security Requirements

The user’s login/signup information should be saved in a secure way on the database with all the required security protections applied, especially in the case of storing the password. The password should be appropriately hashed and salted. The user also should not have any direct access to the database.

## Software Quality Attributes

The primary quality characteristics for the end user are availability, accuracy, reliability, and speed. The end user expects the app to work in a quick, efficient, and easy to use manner.

The primary quality characteristics for the developers are maintainability, accuracy, robustness, interoperability, testability, efficiency, and portability.

The developers need the app to be easy to maintain and capable of functioning well with other components of the system in a testable and verifiable manner.

# Other Requirements

The app must conform with the project guidelines of Ceng 319 at Humber College, and include the required activities, features, and functionality detailed in the course documentation.